A New Simulation Tool For Sensor Networks Based on An Energy-efficient and Fault-tolerant Methodology

Hanene Rouainia[1][0000−0001−7544−988X], Hanen Grichi[2][0000−0002−4601−3574], Laid Kahloul[3][0000−0002−9739−7715], and Mohamed Khalgui[4][0000−0001−6311−3588]

[1] Faculty of Sciences of Tunis, El-Manar University, Tunis, Tunisia
[2] Faculty of Sciences of Bizerte (FSB) - University of Carthage, Bizerte, Tunisia and with School of Electrical and Information Engineering, Jinan University, Zhuhai, China
[3] LINFI Laboratory, Computer Science Department, Biskra University, Biskra, Algeria
[4] Member IEEE, School of Electrical and Information Engineering, Jinan University, Zhuhai, China and with INSAT Institute, University of Carthage, Tunis, Tunisia

Abstract. Recently, reconfigurable wireless sensor networks (RWSNs) have attracted a lot of attention in research and industrial communities. They became more complex and dynamic systems which led to the emergence of many challenges. The lack of energy, real-time constraints, and software and hardware failures are the most important challenges in RWSNs. Indeed, several solutions have proposed to come up with these challenges. To avoid huge costs in terms of money, time, and effort of real experimentation, networks' simulation tools have become an essential necessity to study the impact of the proposed solutions. In this work, we propose a new energy-efficient and fault-tolerant methodology that composed of a set of solutions summarized in the use of mobile sink nodes (MSNs), application of the mobility, resizing, and test packet technique using a multi-agent architecture and an energy-efficient routing protocol. Moreover, we propose a new discrete-event simulation tool named *RWSNSim* designed for sensor networks (WSNs & RWSNs). We present its description, modeling, and provided services. The proposed simulation tool allows simulating sensor networks with and without application of the proposed methodology. Finally, we simulate a case study using *RWSNSim* in a 3D environment which proves the effectiveness of the proposed methodology and demonstrate the efficiency of the suggested simulation tool.

Keywords: Sensor networks · WSNs · RWSNs · Multi-agent architecture · Mobility · Resizing · Mobile sink node · Hardware & Software failures · Test packet technique · Energy-efficient · Fault-tolerant · RWSNSim · Simulation tool

1 Introduction

In recent years, wireless sensor networks (WSNs) have gained worldwide attention in research and industrial communities. They have a wide variety of applications from small-size to large-scale systems and provide unlimited future potentials in a variety of areas such as medical, agricultural, environmental monitoring, smart transportation and intelligent home [22, 47, 5]. WSNs deploy a set of multi-functional and battery-operated devices known as sensor nodes (SNs). They have considerable characteristics

such as low cost, small size, low computing resources, and limited processing. SNs can sense both physical and chemical measurements in the surrounding environment such as temperature, humidity, and gases. They communicate with each other through wireless communication and work cooperatively to transmit a volume of data to a central station, process the sensing data, and execute a set of tasks [1, 38, 39].

In fact, there are several challenges in WSNs such as lack of energy problem, real-time constraints, and hardware/software failures [16, 2]. The mentioned problems occur because of several factors like SNs use limited energy resources, WSNs work under many types of renewable energy resources which are not frequently available, SNs are fragile and prone to failures, communication volume, human effects, and harsh environmental conditions [11, 37, 38].

To resolve the mentioned problems, many solutions and techniques are proposed such as reconfiguration, mobility, resizing of zones, energy-efficient and fault-tolerant routing protocols [42, 17]. Reconfigurable wireless sensor networks (RWSNs) are WSNs with the possibility to execute reconfiguration scenarios thanks to the existence of additional specific devices (i.e., software and hardware agents, mobile SNs, and MSNs) [15, 37, 39].

Indeed, there are several energy-efficient routing protocols in WSNs such as e-NL BEENISH (extended-Network Lifetime Balanced Energy Efficient Network Integrated Super Heterogenous Protocol) [43], WBM-TEEN (Well Balanced with Multi-hops intra-cluster Threshold sensitive Energy Efficient sensor Network protocol) [26], and LEACH (Low Energy Adaptive Clustering Hierarchy) [49]. Otherwise, there are also many fault-tolerant routing protocols for WSNs such as PFTP (Proactive Fault Tolerant Routing Protocol) [32], CFTR (Cluster-based Fault-Tolerant Routing protocol) [25], and FTCM (Fault-Tolerant Clustering-based Multi-path algorithm) [23].

In order to evaluate the proposed solutions and techniques and conclude their impact on the networks' efficiency, the researchers have two choices: real experimentations or simulation. Since the expensive cost, effort, time, and complexity implicated in the real experimentations which associated with the construction and the implementation of WSNs & RWSNs, the developers prefer to get an overview about feasibility and behavior of RWSNs before hardware implementation [39]. Indeed, the analysis and evaluation of the performance of the proposed solutions and techniques through real experimentations are not feasible, complex, and very expensive. As result, to keep up with these challenges, several simulation tools [27] are proposed. They provide many advantages like low cost, giving real-time results, and easy development.

In this work, we present an extended version of the paper [39] with a simplified explanation of the suggested methodology in [38] with some improvements. Briefly, we present an energy-efficient and fault-tolerant methodology which is a unified methodology permits the use of a set of solutions and techniques in energy and time-saving manner to resolve the mentioned problems. The proposed solutions are summarized as follows: i) using a multi-agent architecture which allows to handle the execution of the reconfiguration scenarios, ii) using mobile sink nodes, iii) applying the mobility of mobile entities in a 3D environment which permits to reduce the consumed energy in the network, iv) applying the geographic resizing of zones in a 3D environment which guarantees a good coverage of the largest possible area in the network, and v) using

the test packet technique which permits to detect and isolate the malfunctioning entities in the network [38]. Moreover, we suggest a new discrete-event simulation tool called *RWSNSim* which permits the construction of WSNs and RWSNs, saving them in a database, using two routing protocols (LEACH and WBM-TEEN), plotting the simulation graph, showing the execution report of each monitoring time, drawing the resulting line charts after the simulation, and comparing between the different networks and simulations. The proposed simulation tool permits also to apply the suggested methodology [39, 41].

The rest of this work is organized as follows. After the introduction section, we present the state of the art in section 2. Then, we resume a background about sensor networs, energy issue, and hardware & software failures of networks' entities in Section 3. Morover, the proposed methodology is formalized and detailed in Section 4. Furthermore, the suggested simulation tool is described and reported in Section 5. Section 6 presents a case study simulated using *RWSNSim* to validate and evaluate the performance of the proposed methodology and simulation tool. Finally, the conclusion is drawn in Section 7.

2 State of the Art

The work in [17] proposed an approach based on an intelligent multi-agent distributed architecture using the three forms of reconfiguration (software, hardware, and protocol reconfiguration). The suggested architecture composed of five types of agents: coordinator, supervisor, scheduler, battery manager, and reconfiguration manager. Each agent has specific responsibilities in the system.

In [15], reconfiguration is considered as an efficient and effective solution to resolve the energy problem in WSNs because it makes the WSN satisfy the real-time and energy constraints taking into consideration the system performance optimization. While the paper [10] proposes a new pipelined approach to execute a set of reconfiguration scenarios that need to be applied without altering the performance of the pipeline.

On the other side, the paper [6] proposes a mobile data collection scheme based on the high maneuverability of the unmanned aerial vehicles (UAVs) considering them as MSNs in WSN water monitoring. They transmit data wirelessly to collect monitoring node data efficiently and flexibly. Otherwise, the paper [48] proposes an energy efficient routing schema combined with clustering and sink mobility technologies in WSNs. Otherwise, the authors in [6] propose a new method to find an efficient location service for MSNs using the surplus energy of a solar-powered WSN. The paper [31] studies the energy-efficient routing method with a support for multiple MSNs to effectively alleviate the hot spot problem, as the sink node can move along certain paths, resulting more even distribution of the hot spot nodes. Finally, the authors of [44] propose an intelligent method to discover the optimal path for MSNs using a modified traveling salesman problem (MTSP).

The papers [33, 28] propose the mobility as a solution to minimize the total distance between SNs which permits to decrease the energy consumption to keep the network alive as long as possible. The authors of [3] propose a zone-based sink mobility (ZBSM) approach which permits the mitigation of the energy hole problem and optimal sink

node placement thanks to the zone formation along with controlled sink mobility. To avoid network portioning problems, the sink decides to move toward strongly loaded zone (SLZ) which can be selected using Fuzzy Logic.

Otherwise, the work in [18] suggests the geographic resizing of zones as a solution to the lack of energy and coverage of zones problems in WSNs. Moreover, the paper [9] proposes a data dissemination protocol named hexagonal cell-based data dissemination (HexDD) which exploits a virtual infrastructure and considers dynamic conditions of multiple sinks and sources. The proposed protocol is a fault-tolerant protocol which permits to bypass routing holes created by imperfect conditions of wireless communication in the network. While the authors of [15] tend to apply the resizing of zones and the 3D mobility of mobile SNs into a run-time power-oriented methodology which permits to conserve more energy during the communication among network elements and increase the network lifetime.

On the other hand, many researches suggest energy-efficient routing protocols as effective solutions to the lack of energy problem in WSNs. In fact, there are several energy-efficient routing protocols such as the Balanced Energy Efficient Network Integrated Super Heterogeneous Protocol (BEENISH), extended-Network Lifetime BEENISH (e-NL BEENISH), Information Quality Aware Routing protocol (IQAR), and Low-Energy Adaptive Clustering Hierarchy (WBM-TEEN) [46, 43, 20, 49, 21].

In the same context, there are many fault-tolerant routing protocols which are used to treat the software & hardware failures such as Efficient Fault-Tolerant Multipath Routing Protocol (HDMRP), Proactive Fault Tolerant Routing Protocol (PFTP), and Energy-Efficient and Fault-Tolerant Routing Protocol for Mobile Sensor Network (FTCP-MWSN) [49, 24].

Table 1 presents a discussion about the contribution of some of the mentioned related works and the suggested methodology in this work in terms of the proposed solutions.

Table 1. Discussion about the contribution of some of the mentioned related works and the suggested methodology

Work	MAA	REC	MOB	RES	MSNs	EERP	FTRP&Tech
[15]	✓	✓	✓	✓	✗	✗	✗
[6]	✗	✓	✓	✗	✓	✗	✗
[48, 31]	✗	✓	✓	✗	✓	✓	✗
[3]	✗	✓	✓	✗	✓	✗	✗
[9]	✗	✗	✗	✓	✗	✗	✓
[38]	✓	✓	✓	✓	✓	✓	✓ Only for time-driven routing protocols
Suggested methodology	✓	✓	✓	✓	✓	✓	✓

MAA: multi-agent architecture, **REC**: reconfiguration, **MOB**: mobility, **RES**: resizing, **MSNs**: mobile sink nodes, **EERP**: energy-efficient routing protocol, **FTRP&Tech**: fault-tolerant routing protocol & techniques.

Through table 1, we conclude that the proposed methodology permits the application of several effective solutions in energy and time-saving manner which leads to achieving a high success rate in extending the lifetime of the network compared with other related works.

Indeed, many simulation tools of sensor networks have been proposed by academic and commercial communities such as NS-3, JavaSim, OMNet++, GloMoSiM, and RWiN-Environment [27, 34].

NS-3 (Network Simulator-3) [29] is a discrete-event simulator designed for Internet systems. It was launched in June 2008 as an open-source project and licensed under the GNU GPLv2 license. It was targeted primarily at research and educational uses and maintained by a worldwide community. The latest version of NS-3 is NS-3.35 which provides several improvements such as IPv6 support for NixVectorRouting and a group mobility helper. It is released in October 1, 2021.

J-Sim (JavaSim) simulator [19] is a general purpose simulator started in year 1997 and used by various commercial and academic organizations. It is an object-oriented simulation package based upon C++SIM. J-Sim permits executing several script-based languages such as Python and Perl thanks to the use of a script interface. It provides a framework for WSN simulation using INET, ACA, and Wireless Protocol stack. The latest version of J-Sim is 2.2.0 GA which is released in January 4, 2020.

OMNet++ [30] is a powerful object-oriented discrete-event simulator which was launched in September 1997 and has a large number of users in educational, academic, and research-oriented commercial institutions. It is designed for the simulation of distributed and parallel systems, computer networks, performance evaluation of complex software systems, and modeling of multiprocessors. It is a modular, extensible, and component-based open-architecture simulation framework. The current version of OMNet++ is 5.7 which is released in October 6, 2021.

GloMoSiM (Global Mobile Information System) simulator [12] is a discrete-event scalable simulation software developed by Parallel Computing Lab at UCLA. It is designed for large-scale wireless network systems using parallel discrete-event simulation provided by Parsec. It supports three communication protocols: traditional Internet protocols, multi-hop wireless communication, and direct satellite communication. GloMoSiM uses Parsec compiler to compile the simulation of protocols. It allows the simulation of networks which contain thousands of heterogeneous nodes and communication links.

RWiN-Environment (Reconfigurable Wireless Network Environment) [40] is a graphical tool developed in the research laboratory LISI at the University of Carthage in Tunisia. It is develop for the modeling, formal verification, validation, simulation, code generation and implementation of RWSNs. It also permits evaluating the services of the RWiN-Methodology which is proposed to construct, analyse, develop, and verify an RWSN system in order to reduce the consumed energy by the network elements [14]. It is intended for a large community that allows RWSN designers to develop safety systems. RWiN-Environment is able to treat all forms of reconfiguration together and simulate and validate reconfigurable systems in the same environment, reduce the formal verification time, deploy the obtained code generation on STM32F4 microcontrollers,

and save the obtained graphics which allows the study of the history of all RWSN implementations [13].

Table 2 presents a comparison of the mentioned simulation tools according to general parameters.

Table 2. Comparison of some simulation tools [39]

Name	Type	Prog. Lang	License	GUI	Scalability	Portability	Designed for (WSNs or RWSNs)
NS-3	Discrete event	C++, Python	Open source	Limited	Large	Yes	WSNs
JavaSim	Discrete event	Java	Open source	Medium	Medium	Yes	WSNs
OMNet++	Discrete event	C++	Open source	Excellent	Large	Yes	WSNs
GloMoSiM	Discrete event	C	Open source	Limited	Large	Yes	WSNs
RWiN-Environment	Discrete event	Java	Open source	Excellent	Medium	Yes	RWSNs
RWSNSim	Discrete event	Java	Open source	Excellent	Medium	Yes	WSNs, RWSNs

Through table 2, we remark that the most obvious strength of the proposed simulation tool is its ability to simulate both WSNs and RWSNs. For that, the proposed simulation tool *RWSNSim* is considered as the first simulation tool designed to simulate both WSNs and RWSNs. Moreover, it has also other strengths summarized as follows: license, graphical user interface (GUI), and portability.

3 Background

We present in this section a semi-formal description of the components and architecture of sensor networks. Then, we describe the energy issue by formalizing the energy model and problem. Finally, we detail the software & hardware problems in sensor networks [38].

3.1 WSN Architecture

A wireless sensor network *(W)* in a 3D environment is composed of:

1. A base station *BS*,
2. A set of zones formalized by: $S_Z(W) = \{ \bigcup_{k=1}^{Nb_Z(W)} \{Z_k\} \}$ where $Nb_Z(W)$ is the number of zones in W and Z_k is a zone in W,

3. Each zone Z_k contains:
   - A gateway G_k,
   - A set of fixed sensor nodes formalized by: $S_N(Z_k) = \{ \bigcup_{i=1}^{Nb_N(Z_k)} \{N_{i,k}\}\}$, where $N_{i,k}$ is a sensor node in Z_k and $Nb_N(Z_k)$ is the total number of sensor nodes in Z_k.

Fig. 1 shows the architecture of a WSN.

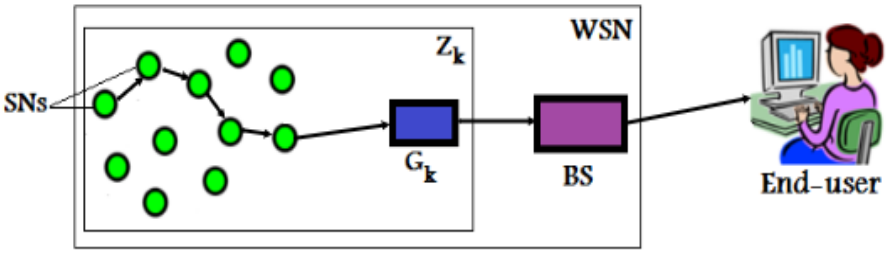

Fig. 1. WSN Architecture

Each sensor node in *W* has a sensing unit composed of: *i)* a set of sensors formalized by $S_{Sens}^{N_{i,k}} = \{Sens_{j,N_{i,k}} | \ i \in [1..Nb_N(Z_k)], \quad k \in [1..Nb_Z(W)] \quad and \quad j \in [1..Nb_{Sens}(N_{i,k})]\}$ where $Sens_{j,N_{i,k}}$ is a sensor in $N_{i,k}$ and $Nb_{Sens}(N_{i,k})$ is the total number of sensors in $N_{i,k}$ and *ii)* an analog to digital converter (ADC). These sensors are designed for sensing the chemical and physical conditions in the surrounding environment such as temperature, gases, humidity, light, luminosity, pressure, etc. It contains also a transceiver unit connected with an antenna, a processing unit composed of a microcontroller, and an external memory. Each sensor node contains also a power unit composed of two batteries. The first one is the principal battery $B_{pr}(N_{i,k})$ and the second one is the additional battery $B_{add}(N_{i,k})$. The principal battery is rechargeable by the additional battery and this last one is rechargeable from the harvesting energy. Finally, to generate the harvesting energy, each sensor node has a power generator connected with the harvesting energy module [35, 4, 45, 36, 32].

Each entity E in W has three coordinates $\{(x_E, y_E, z_E) | E \in \{S_N(W), S_G, BS\}\}$, where $S_N(W)$ is the set of sensor nodes in W and S_G is the set of gateways in W. These coordinates represent the position of E in W.

3.2 RWSN Architecture

A reconfigurable wireless sensor network *(R)* in a 3D environment is composed of:

1. A base station *BS*,
2. A controller agent Ag_{Ctrl}

3. A set of zones formalized by: $S_Z(R) = \{ \bigcup_{k=1}^{Nb_Z(R)} \{Z_k\}\}$ where $Nb_Z(R)$ is the number of zones in R and Z_k is a zone in R,
4. Each zone Z_k contains:
   - A zone agent Ag_k,
   - A set of MSNs formalized by $S_{SN}(Z_k) = \{SN_{m,k} |\ k \in [1..Nb_Z]\ m \in [1..Nb_{SN}(Z_k)]\}$, where $SN_{m,k}$ is a mobile sink node in Z_k and $Nb_{SN}(Z_k)$ is the number of MSNs in Z_k,
   - A set of sensor nodes formalized by: $S_N(Z_k) = \{ \bigcup_{i=1}^{Nb_N(Z_k)} \{N_{i,k}\}\}$, where $N_{i,k}$ is a sensor node in Z_k and $Nb_N(Z_k)$ is the total number of SNs in Z_k,
   - There are two types of SNs: fixed and mobile ones. The fixed SNs are formalized by $S_{FN}(Z_k) = \{N_{i,k} |\ k \in [1..Nb_Z]\ \ i \in [1..Nb_N(Z_k)]\}$. The mobile SNs are formalized by $S_{MN}(Z_k) = \{N_{i,k} |\ k \in [1..Nb_Z]\ \ i \in [1..Nb_N(Z_k)]\}$. We denote that $S_{MN} \cup S_{FN} = S_N \ \ and \ \ S_{MN} \cap S_{FN} = \emptyset\}$.

Fig. 2 displays the architecture of an RWSN.

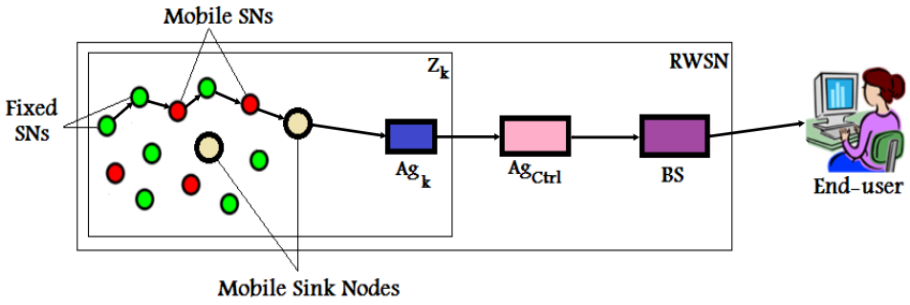

Fig. 2. RWSN Architecture

Indeed, the fixed sensor nodes in R are the same in W. Otherwise, the mobile SNs contain the same components as fixed SNs with two additional components: a location finder and a mobilizer. The location finder is used to identify the position of the mobile SNs. While the mobilizer permits them to move easily in the network.

Otherwise, MSNs have the same components as mobile SNs with two high-charge level batteries and without the sensing unit. They play the role of gateway between sensor nodes and zone agents [45, 36].

Each fixed entity E_f in R has three coordinates formalized by $\{(x_{E_f}, y_{E_f}, z_{E_f}) \mid E_f \in \{S_{FN}(R), S_{Ag_k}, Ag_{Ctrl}, BS\}\}$, where S_{Ag_k} is the set of zone agents in R. These coordinates represent the position of E_f in R. While each mobile entity E_m in R has three original coordinates formalized by $\{(x_{E_m}, y_{E_m}, z_{E_m}) | E_m \in \{S_{MN}(R), S_{SN}\}\}$. These coordinates represent the original position of E_m in R. Each mobile entity E_m in R has also three actual coordinates formalized by $\{(x'_{E_m}, y'_{E_m}, z'_{E_m}) | E_m \in \{S_{MN}(R), S_{SN}\}\}$ which represent the actual position of E_m after its last executed mobility task.

Finally, the charge capacity of each entity E in W and R is formalized by $capacity(E) = capacity(B_{pr}(E)) + capacity(B_{add}(E))$. While the total charge of each entity E in W and R is formalized by $C(E) = C(B_{pr}(E)) + C(B_{add}(E))$.

3.3 Energy Model in RWSNs

In each period ρ, each entity E in R executes a set of tasks which is formalized by $T_\rho(E) = \{\tau_1, \tau_2, ..., \tau_{Nb_\tau(E)}\}$, where $Nb_\tau(E)$ is the total number of tasks that executed by E during ρ. Each task τ_a is associated with a trilogy formalized by $Tr_{\rho,\tau_a}(E) = \{t_{exec}(\tau_a), e_c(\tau_a), p_{\tau_a}\}$ such that $t_{exec}(\tau_a)$ is the execution time of τ_a, $e_c(\tau_a)$ is the energy consumed by E to execute τ_a, and p_{τ_a} is a function formalized by:

$$p_{\tau_a} = \begin{cases} n \ \ if \ \ \tau_a \ \ is \ \ executed \ \ n \ \ times \\ 0 \ \ if \ \ not \end{cases} \tag{1}$$

Indeed, we can predict the approximate value of the consumed energy by each entity E in R during ρ using the following formula:

$$EC_{[\rho]}(E) = \int_{t_x}^{t_y} \sum_{a=1}^{Nb_\tau^E} (p_{\tau_a}(E) \times e_c(\tau_a(E)))dt + \epsilon \tag{2}$$

where $\rho = |t_y - t_x|$, Nb_τ^E is the number of tasks executed by E during the period ρ.

Otherwise, we can predict also the approximate value of the produced energy by the additional battery in each entity E in R during ρ using the following formula:

$$EP_{[\rho]}(E) = \int_{t_x}^{t_y} \sum_{t_i}^{t_j} [e_{prod} \times (t_j - t_i)] \tag{3}$$

where e_{prod} is the produced energy in each time unit, $t_x \leq t_i \leq t_y$,$t_x \leq t_j \leq t_y$ and $\rho = |t_y - t_x|$.

3.4 Energy Problem in Sensor Networks

In fact, sensor networks are designed for specific applications which range from small-size healthcare surveillance systems to large-scale environmental monitoring systems. In recent years, several challenges appear in sensor networks where the energy problem is the most important challenge [38].

The lack of energy problem in sensor networks is a disturbing problem, especially in crucial domains such as medical, nuclear, and the prediction of natural disasters. Indeed, sensor networks operate using renewable energy resources which are characterized by the oscillating presence in the surrounding environment such as solar energy, wind energy, hydropower, and bioenergy. For that, we explain the energy problem in the following.

We assume that the energy production times can interfere with the energy consumption times. But in several periods renewable energy cannot be available. For that,

we consider that the harvesting energy is unavailable in the period $\rho = t_b - t_a$ which can be a long period. Therefore, the energy produced by the harvesting energy module in each entity E in the network is almost equal to zero ($EP_\rho \approx 0$). While the energy consumed by the active entities in the network increases with time ($EC_\rho >> 0$) (Fig. 3). As a result, the total charge of the remaining active entities will reach the threshold β which is a threshold for the energy charge in each entity E in the network and it is used to deactivate the entity E ($C(E) \leq \beta$).

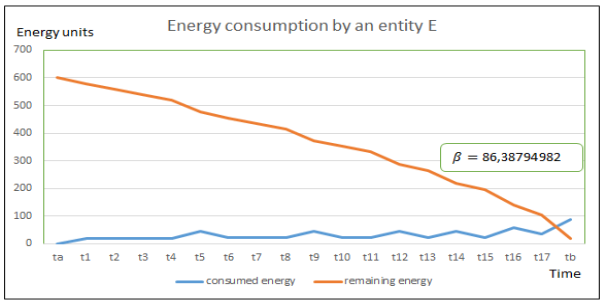

Fig. 3. Energy consumption by an entity E

As more time passes, the distance between the remaining active entities in the network will be expanded which leads to increasing the consumed energy by these entities which speed up their deactivation. As a result, more entities will be deactivated and the network can stop working until human intervention or the return of harvesting energy.

3.5 Software & Hardware Failure Problems

In fact, wireless sensor networks are mainly composed of a set of sensor nodes that are fragile and prone to many failures such as software & hardware failures, unreliable wireless connections, and malicious attacks. These failures make the network performance degrade significantly during its lifespan. The probability of sensor node failures increases because of several factors such as increasing the number of sensor nodes and deploying them in dangerous environments. SNs may fail because of many reasons such as lack of energy and failure of one of the sensor node components. They may also sense and transmit incorrect data. Otherwise, the network topology can be affected because of link failures which may cause some delays in data communication. In WSNs, every failure affects the efficiency of the network by disrupting the execution of the data transmission process and reconfiguration scenarios. For that, fault detection in WSNs is an efficient solution that has to be precise to avoid negative alerts, and rapid to limit loss. In this work, we treat the software/hardware failures that disrupt the receiving and sending tasks executed by the different network entities. We use a test packet technique that permits detecting the malfunctioning entities in the network and isolating them [50, 7, 8].

4 New Energy-Efficient and Fault-Tolerant Methodology In RWSNs

In this section, we formalize a set of constants, variables, and functions that are used to describe and treat the energy problem and the hardware & software failures. Then, we formalize the different rules that are used in the suggested methodology to regulate the application of the proposed solutions and techniques.

4.1 Terminology

Table 3 presents the most executed tasks by the entities in RWSNs.

Table 3. Set of the most energy-consuming tasks executed by different entities in *R*

E	$\tau_a(E)$	$Description$
$E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$	$\tau_{Recept}(E)$	Receiving packets from predecessors.
	$\tau_{Send}(E)$	Sending packets to successors.
$Ag \in S_{Ag}$	$\tau_{Deact}(Ag)$	Deactivate the entity E which is controlled by Ag.
	$\tau_{Act}(Ag)$	Activate the entity E which is controlled by Ag.
Ag_{Ctrl}	$\tau_{Res}(Ag_{Ctrl})$	Apply the resizing of zones.
	$\tau_{Iso}(Ag_{Ctrl})$	Isolate the malfunctioning zone agent Ag_k.
Ag_k	$\tau_{Mob}(Ag_k)$	Apply the mobility of mobile entities.
	$\tau_{Iso}(Ag_k)$	Isolate the malfunctioning entity E where $E \in \{S_N, S_{SN}\}$.
	$\tau_{Org}(Ag_k)$	Organize the sensor nodes in Z_k in subzones into clusters.
$E \in \{S_{SN}, S_{MN}\}$	$\tau_{Move}(E)$	Moving to another position in Z_k.
$N_{i,k} \in S_N$	$\tau_{Sens}(N_{i,k})$	Sensing the physical and chemical conditions of the surrounding environment by $Sens_{j,N_{i,k}}$.

Firstly, we propose a set of constant and variable thresholds and a set of functions to describe the energy problem in RWSNs. They are summarized as follows:

– α_E**:** is a constant threshold for the energy charge in each entity E in the network where $E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$. It is used by agents to activate the inactive entities in the network. While, $\alpha_{N_{i,k}}$ is used by zone agents to determine the poor SNs in their zones in terms of energy charge and to apply the mobility. α_E is formalized by:

$$0.15 \times capacity(E) \leq \alpha_E \leq 0.2 \times capacity(E) \tag{4}$$

– β_E**:** is a variable threshold for the energy charge in each entity E. It is used by the agent that controls E to deactivate it. It is formalized by:

$$\beta_E = EC_{[\rho]}(E) \tag{5}$$

where $E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$ and ρ is the period that is required for one monitoring time.

– γ**:** is a constant threshold for the number of active SNs in each zone Z_k. It is used by zone agents to apply the mobility of mobile SNs. It is formalized by:

$$0.3 \times Nb_N(Z_k) \leq \gamma \leq 0.4 \times Nb_N(Z_k) \tag{6}$$

– λ**:** is a constant threshold for the number of active SNs in each zone Z_k. It is used by Ag_{Ctrl} to apply the resizing of zones. It is formalized by:

$$0.15 \times Nb_N(Z_k) \leq \lambda \leq 0.2 \times Nb_N(Z_k) \tag{7}$$

– $state(E)$**:** is a boolean function that indicates the state of the entity E. It is formalized by:

$$state(E) = \begin{cases} 1 \ if \ \ E \ \ is \ \ active \\ 0 \ \ if \ \ not \end{cases} \tag{8}$$

where $E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$

– $isFree(E)$**:** is a boolean function that indicates the recent movements of a mobile entity E. It is formalized by:

$$isFree(E) = \begin{cases} 0 \ \ if \ \ E \ \ has \ \ moved \ \ recently \ \ as \ \ close \ \ to \\ \qquad a \ \ poor \ \ sensor \ \ node \ \ N_{a,k} \\ 1 \ \ if \ \ not \end{cases} \tag{9}$$

where $E \in \{S_{SN}, S_{MN}\}$

– $isMalFun(E)$**:** is a boolean function that indicated if the entity E is a malfunctioning entity or not. It is formalized by:

$$isMalFun(E) = \begin{cases} 1 \ \ if \ \ E \ \ is \ \ aMalfunctioning \ \ entity \\ 0 \ \ if \ \ not \end{cases} \tag{10}$$

where $E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$

– $EPres(N_{i,k})$**:** is a boolean function that indicates if the energy problem of a poor sensor node $N_{i,k}$ is resolved recently. It is formalized by:

$$EPres(N_{i,k}) = \begin{cases} 1 \ \ if \ \ the \ \ energy \ \ problem \ \ of \ \ N_{i,k} \\ \qquad is \ \ resolved \ \ recently \\ 0 \ \ if \ \ not \end{cases} \tag{11}$$

– $Nb_{Act}(N_{i,k}, Z_k)$**:** is the total number of active SNs in a zone Z_k. It is formalized by:

$$Nb_{Act}(N_{i,k}, Z_k) = \sum_{i=1}^{Nb_N(Z_k)} state(N_{i,k}) \tag{12}$$

– $Nb_{Act}(SN_{m,k}, Z_k)$**:** is the total number of active MSNs in a zone Z_k. It is formalized by:

$$Nb_{Act}(SN_{m,k}, Z_k) = \sum_{i=1}^{Nb_{SN}(Z_k)} state(SN_{m,k}) \tag{13}$$

Moreover, we propose a set of variables that are used to detect the malfunctioning entities in the network and isolate them. They are summarized as follows:

- δ**:** is a percentage factor that must be defined by the administrator of the network to fix the waiting time to receive the sensing data in time-driven routing protocols or the acknowledge messages in both time-driven and event-driven routing protocols.
- $t_{rep_1}(E)$**:** is the response time of the entity E when receiving a request for sensing data.
- $dl_1(E_1, E_2)$: is a deadline for the receiving task when E_2 sends a request for sensing data from E_1.
- $t_{rep_2}(E_1)$**:** is the response time of the entity E_1 when receiving a test packet from E_2.
- $dl_2(E_1, E_2)$: is a deadline for the receiving task when E_2 sends a test packet to E_1 and waits for the acknowledge message. It is used by E_2 to accurately detect the malfunctioning entity.

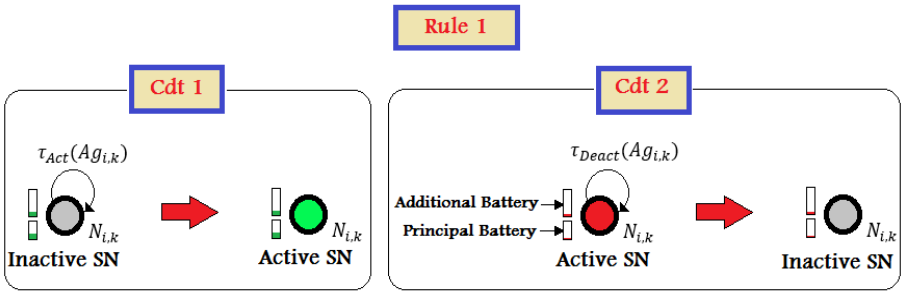

Fig. 4. Illustration diagram of an example of **Rule 1**

Otherwise, we suggest a set of variables that permit to Ag_{Ctrl} and BS to send alert messages and call human intervention. They are summarized as follows:

- $Nb_{flrN}(Z_k)$**:** is the number of malfunctioning SNs in Z_k.
- $Nb_{flrSN}(Z_k)$**:** is the number of malfunctioning sink nodes in Z_k.
- $Nb_{flrAg}(R)$**:** is the number of malfunctioning zone agents in R.
- $flrN$**:** is a constant threshold for the number of malfunctioning SNs in Z_k. It's value is formalized by $flrN \geq 0.1 \times NbN(Z_k)$.
- $flrSN$**:** is a constant threshold for the number of malfunctioning MSNs in Z_k. It's value is formalized by $flrSN \geq 0.2 \times NbSN(Z_k)$.
- $flrAg$**:** is a constant threshold for the number of malfunctioning zone agents in R. It's value is formalized by $flrAg \geq 0.1 \times NbZ$.

4.2 Rules

Indeed, we propose a set of rules to distribute the different responsibilities in the network, regulate the application of different reconfiguration scenarios, and realize the

proposed solutions and techniques in the suggested methodology. They are formalized as follows:

– **Rule 1:** Activation & Deactivation of the network entities (Fig. 4).
  • **Cdt 1:** $if\ (C(E) \geq \alpha_E\ \&\ state(E) = 0\ \&\ isMalFun(E) = 0)\ then$ the agent Ag that controls the entity E decides to activate it. As a result, the following task must be executed by Ag:

$$\tau_{Act}(Ag) = Activate(E)$$

  where $Ag \in S_{Ag}$ and $E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$.
  • **Cdt 2:** $if\ (C(E) \leq \beta_E\ \&\ state(E) = 1)\ then$ the agent Ag that controls the entity E decides to deactivate it. As a result, the following task must be executed by Ag:

$$\tau_{Deact}(Ag) = Deactivate(E)$$

  where $Ag \in S_{Ag}$ and $E \in \{Ag_{Ctrl}, S_{Ag_k}, S_{SN}, S_N\}$.

– **Rule 2:** Application of mobility (Fig. 5).
  • **Cdt 1:** $if\ \ (state(N_{i,k}) = 1\ \ \&\ \ C(N_{i,k}) \leq \alpha_{N_{i,k}}\ \ \&\ \ EPres(N_{i,k}) = 0\ \ \&\ \ Nb_{Act}(N_{i,k}, Z_k) > \gamma\ \&\ N_{i,k} \in SZ_{m,k})\ then\ Ag_k$ decides to apply the mobility of $SN_{m,k}$ taking into consideration the following subcondition:
    ∗ **Cdt 1.1:** $if\ (isFree(SN_{m,k}) = 1)\ then\ Ag_k$ and $SN_{m,k}$ must execute the following tasks:

$$\tau_{Mob}(Ag_k) = ApplyMobility(SN_{m,k})$$
$$\tau_{Move}(SN_{m,k}) = MoveTo(NewPosition)$$

  • **Cdt 2:** $if\ \ (state(N_{i,k}) = 1\ \ \&\ \ C(N_{i,k}) \leq \alpha_{N_{i,k}}\ \ \&\ \ EPres(N_{i,k}) = 0\ \ \&\ \ Nb_{Act}(N_{i,k}, Z_k) \leq \gamma\ \&\ N_{i,k} \in SZ_{m,k})\ then\ Ag_k$ decides to apply the mobility according to the following subconditions:
    ∗ **Cdt 2.1:** $if\ |N_{i,k}SN_{m,k}| \leq |N_{i,k}E|\ then\ Ag_k$ decides to apply the mobility considering the following cases:
      · *Case 1:* $if\ (isFree(SN_{m,k}) = 1)\ then\ Ag_k$ applies the mobility of $SN_{m,k}$ according to **Cdt 1.1** without considering **Cdt 1**.
      · *Case 2:* $if\ (isFree(SN_{m,k}) = 0)\ then\ Ag_k$ applies the mobility of $N_{a,k}$ according to **Cdt 2.2** without considering **Cdt 2**.

    such that $E \in \{SN_{m,k}, N_{a,k}\}$ where $N_{a,k}$ is the closest active mobile SN to $N_{i,k}$
    ∗ **Cdt 2.2:** $if\ (|N_{i,k}SN_{m,k}| > |N_{i,k}N_{b,k}|\ then\ Ag_k$ applies the mobility taking into consideration the following cases;
      · *Case 1:* $if\ (isFree(N_{a,k}) = 1\ \&C(N_{a,k}) > [\int_{t_x}^{t_y}(e_c(\tau_{Move}(N_{a,k}))dt + \alpha_{N_{a,k}} + \epsilon)])\ then\ Ag_k$ applies the mobility of $N_{a,k}$, where:

$$e_c(\tau_{Move}(N_{a,k})) = (|N_{i,k}N_{a,k}| - |N_{i,k}SuccN_{i,k}|/2) \times e_{Mob}$$

      where e_{Mob} is the energy consumed by $N_{a,k}$ to move one meter.

Therefore, Ag_k and $N_{a,k}$ must execute the following tasks:

$$\tau_{Mob}(Ag_k) = ApplyMobility(N_{a,k})$$
$$\tau_{Move}(N_{a,k}) = MoveTo(NewPosition)$$
$$\tau_{Org}(Ag_k) = OrganizeNodes()$$

case 2: $if\ (isFree(N_{a,k} = 0)\,||\,C(N_{a,k}) \leq [\int_{t_x}^{t_y}(e_c(\tau_{Move}(N_{a,k}))dt + \alpha_{N_{a,k}} + \epsilon)])\ then\ Ag_k$ decides to apply the mobility of $SN_{m,k}$ according to **Cdt 1** without considering its conditions.

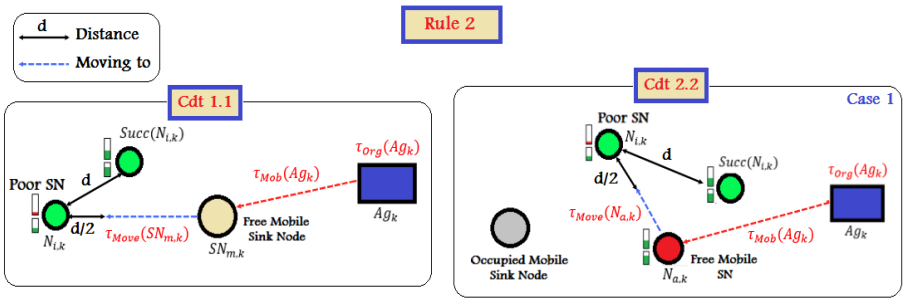

Fig. 5. Illustration diagram of examples of **Rule 2**

– **Rule 3:** Application of the resizing of zones (Fig. 6).
  - **Cdt 1:** $if\ (Nb_{Act}(N_{i,k}, Z_k) \leq \lambda)\ then\ Ag_{Ctrl}$ decides to apply the resizing of zones of Z_k and its neighbor zone Z_a which contains the minimum number of active SNs and/or active MSNs. As a result, Ag_{Ctrl} and Ag_k must execute the following tasks:

$$\tau_{Res}(Ag_{Ctrl}) = Resizing(Z_k, Z_a)$$
$$\tau_{Deact}(Ag_k) = Deactivate(Ag_k)$$
$$\tau_{Org}(Ag_a) = OrganizeNodes()$$

– **Rule 4:** Detection and isolation of malfunctioning entities in the network (Fig. 7).
  - **Cdt 1:** $if\ (t_{rep_1}(N_{i,k}) > dl_1(N_{c,k}, SN_{m,k}))\ then\ SN_{m,k}$ sends test packets to the sensor nodes belonging to the cluster whose head is $N_{c,k}$ from closest to farthest until finding the malfunctioning one.
    * **Cdt 1.1:** $if\ (t_{rep_2}(N_{i,k}) > dl_2(N_{i,k}, SN_{m,k}))\ then\ SN_{m,k}$ sends an alert message to Ag_k to inform it that $N_{i,k}$ is a malfunctioning sensor node. As a result, Ag_k isolates $N_{i,k}$. Therefore, Ag_k must execute the following tasks:

$$\tau_{Iso}(Ag_k) = Isolate(N_{i,k})$$
$$\tau_{Org}(Ag_k) = OrganizeNodes()$$

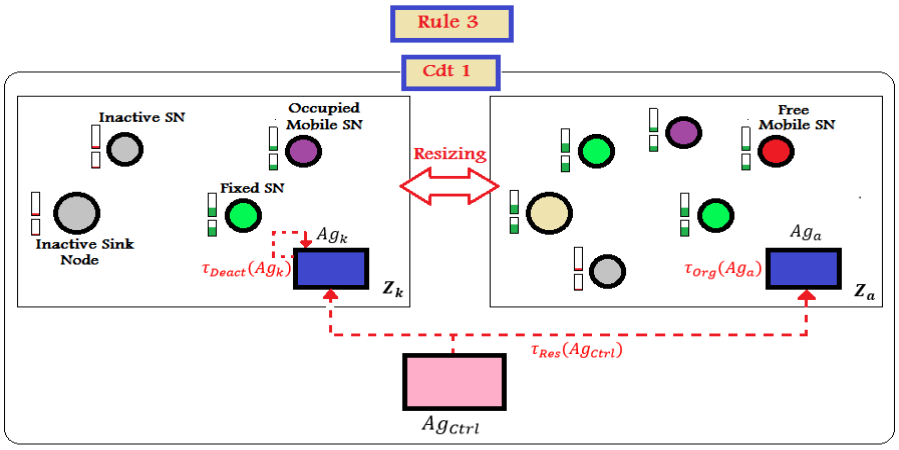

Fig. 6. Illustration diagram of an example of **Rule 3**

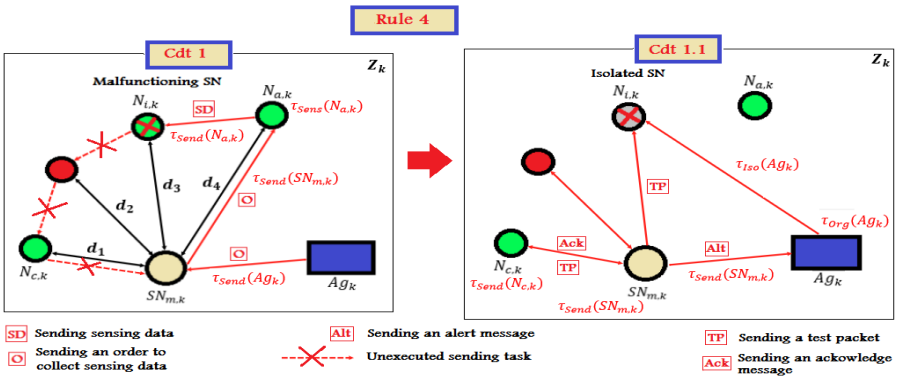

Fig. 7. Illustration diagram of an example of **Rule 4**

- **Cdt 2:** $if\ (t_{rep_1}(SN_{m,k}) > dl_1(SN_{m,k}, Ag_k))\ then\ Ag_k$ sends a test packet to $SN_{m,k}$ to detect if it is a malfunctioning sink node.
  - **Cdt 2.1:** $if\ (t_{rep_2}(SN_{m,k}) > dl_2(SN_{m,k}, Ag_k))\ then\ Ag_k$ isolates $SN_{m,k}$. As a result, Ag_k must execute the following tasks:

$$\tau_{Iso}(Ag_k) = Isolate(SN_{m,k})$$
$$\tau_{Org}(Ag_k) = OrganizeNodes()$$

- **Cdt 3:** $if\ (t_{rep_1}(Ag_k) > dl_1(Ag_k, Ag_{Ctrl}))\ then\ Ag_{Ctrl}$ sends a test packet to Ag_k to detect if it is a malfunctioning zone agent.
  - **Cdt 3.1:** $if\ (t_{rep_2}(Ag_k) > dl_2(Ag_k, Ag_{Ctrl}))\ then\ Ag_{Ctrl}$ isolates Ag_k and applies the resizing of zones (**Rule 3**). As a result, Ag_{Ctrl} must execute

the following tasks:

$$\tau_{Iso}(Ag_{Ctrl}) = Isolate(Ag_k)$$
$$\tau_{Res}(Ag_{Ctrl}) = Resizing(Z_k, Z_a)$$

Note: **Rule 4** is used in time-driven routing protocols. While, in event-driven routing protocols, test packets are sent periodically from Ag_{Ctrl} to all SNs, passing through zone agents and MSNs taking into consideration **Cdt 1.1, Cdt 2.1,** and **Cdt 3.1** to detect and isolate the malfunctioning entities without stopping when detecting a malfunctioning entity.

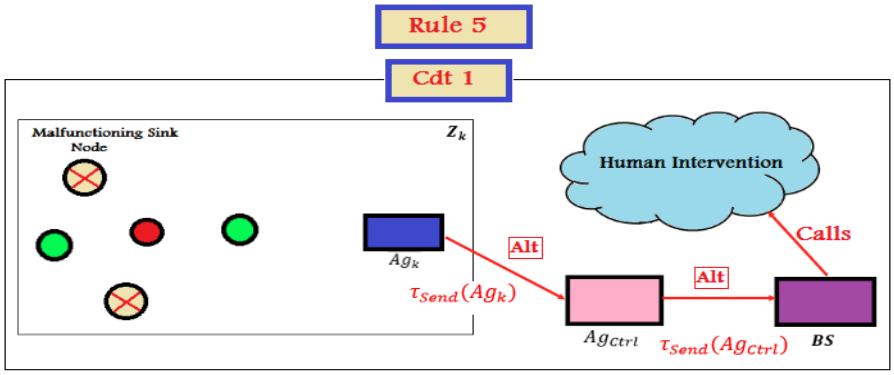

Fig. 8. Illustration diagram of an example of **Rule 5**

– **Rule 5:** Resolve the hardware & software failure problem by calling human intervention (Fig. 8).

- **Cdt 1:** $if\ (Nb_{flrN}(Z_k) \geq flrN\ ||\ Nb_{flrSN}(Z_k) \geq flrSN)\ then\ Ag_k$ sends an alert message to Ag_{Ctrl} which transmits it to BS which calls human intervention.
- **Cdt 2:** $if\ (Nb_{flrAg}(R) \geq flrAg)\ then\ Ag_{Ctrl}$ sends an alert message to BS which calls human intervention.

5 RWSNSim: Reconfigurable Wireless Sensor Networks Simulator

In this section, we describe the suggested simulation tool and present its provided services. Furthermore, we model the used databases to store the information of sensor networks using entity relationship diagrams [39]. Finally, we present the used algorithms to apply the proposed methodology and to develop the suggested simulation tool [38].

5.1 Description

The proposed simulator *RWSNSim* is a discrete-event simulation tool designed for WSNs and RWSNs. It is developed using Java for research and educational use. It is a free, open-source, extensible, and reusable simulation tool. *RWSNSim* supports several operating systems such as Windows and UNIX. It is capable of simulating sensor networks that contain hundreds of entities. It is a desktop simulator that allows constructing WSNs and RWSNs and simulating them using two routing protocols (LEACH and WBM-TEEN) with and without the proposed methodology.

The suggested simulation tool provides several services during the simulation process, including the following:

- Create new WSNs and RWSNs with two manners: manual & regular and automatic & random.
- Save the created sensor networks in a database using *MySQL* or *hsqldb* according to the user's choice.
- Provide two routing protocols: LEACH and WBM-TEEN.
- Display the simulation graph of each simulation using *jgraph*, *jgraphx*, and *jgrapht* libraries.
- Show the execution report for each monitoring time during the simulation.
- Extract the simulation results and display them in form of line charts using *jfreechart* library.
- Comparison of different networks and simulations.

5.2 Database Modeling

Indeed, *RWSNSim* uses databases to store information related to the created sensor networks. To make these databases, we use entity relationship diagrams (ERDs) to model them. In fact, we have two ERDs that are explained as follows:

- *ERD for WSNs:* Fig. 9 displays the ERD of created WSNs in *RWSNSim* which is used to generate the databases used to store information related to the created WSNs that can be entered by the user (manual & regular manner) or generated automatically (automatic & random manner). Indeed, this ERD contains the following databases:
  - **[DB:WSNs]:** is a database that includes a single weak entity named *WSN_Infos* which contains the general and common information related to each WSN such as its name (*WSN_Name*) which is the key attribute, the number of sensor nodes (*NbNodes*), and the used routing protocol (*protocol*).
  - **[DB:WSN_Name]:** For each constructed WSN with *RWSNSim*, a database with the same name is created to store its information. It contains three strong entities that are summarized as follows:
    - **Zone:** includes the common information of each zone in the network such as its state (*State*), and the number of SNs that belong to it (*NbNodes*).
    - **Gateway:** includes the general information of each gateway in the network like its name (*Name*) and the coordinates of its position in the network (*Coordinates*).

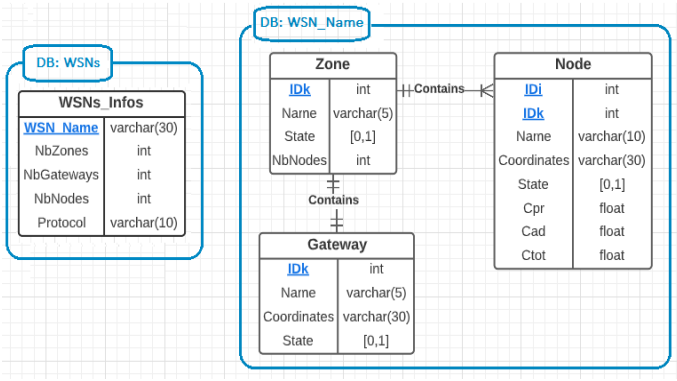

Fig. 9. Entity relationship diagram for WSNs [39]

∗ **Node:** comprises the main information of each sensor node in the network such as two identifiers (*IDi*, *IDk*) which are the key attributes of the entity and its initial total charge (*Ctot*).

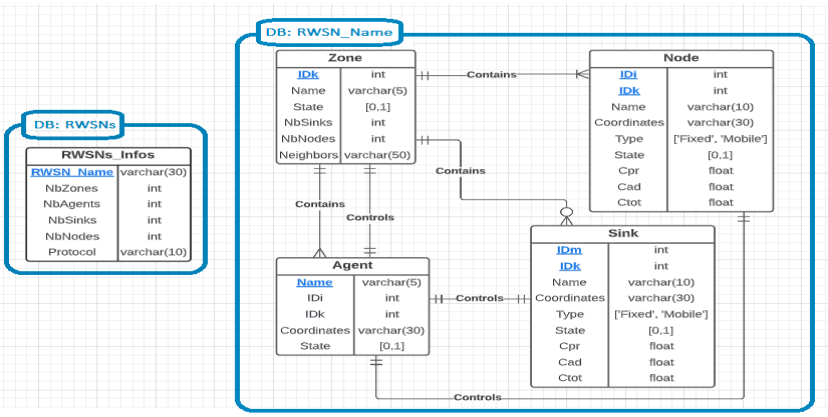

Fig. 10. Entity relationship diagram for RWSNs [39]

– *ERD for RWSNs:* Fig. 10 displays the ERD of created RWSNs in *RWSNSim* which is used to generate the databases used to store information related to the created RWSNs that can be entered by the user (manual & regular manner) or generated automatically (automatic & random manner). In fact, this ERD contains the following databases:
  - **[DB:RWSNs]:** is a database that includes a single weak entity named $RWSN\_Infos$ which contains the general and common information that related to each created RWSN such as the number of zones (*NbZones*), the number of agents (*NbAgents*), and the number of MSNs (*NbSinks*).

- **[DB:RWSN_Name]:** For each constructed RWSN with *RWSNSim*, a database with the same name is created to store its information. It contains four strong entities that are defined as follows:
  - **Agent:** includes the general information of each agent in the network such as its identifiers (*IDi, IDk*), the coordinates of its initial position in the network (*Coordinates*), and its state (*State*).
  - **Zone:** comprises the common information of the network zones like the number of MSNs that belong to it (*NbSinks*), the number of SNs that belong to it (*NbNodes*), and the list of its neighbor zones (*Neighbors*).
  - **Sink:** contains the general information of each sink node in the network such as its type which can be fixed or mobile (*Type*), the coordinates of its initial position in the network (*Coordinates*), and its initial total charge (*Ctot*).
  - **Node:** comprises the common information of SNs in the network like two identifiers (*IDi, IDk*) which are the key attributes, its type which can be fixed or mobile (*Type*), its initial charge of the principal battery (*Cpr*), and its initial charge of the additional battery (*Cad*).

5.3 Algorithms

In order to develop *RWSNSim*, we use many algorithms to apply the suggested methodology. The zone agent Ag_k must execute **Algorithm 1** to apply the mobility in Z_k taking into consideration the **Rule 2** conditions. Moreover, the controller agent Ag_{Ctrl} must execute **Algorithm 2** to apply the resizing of zones according to the **Rule 3** conditions. Finally, the controller agent Ag_{Ctrl} must execute **Algorithm 3** to detect and isolate the malfunctioning zone agents (for time-driven routing protocols (*Ex:* LEACH)).

Algorithm 1: Apply the mobility in Z_k [38]

Require: Set of SNs and MSNs.
Ensure: Minimize the distance between SNs and MSNs.
for $i = 0$ *to* $VSZ.size()$ **do**
  for $j = 0$ *to* $VSZ.get(i).vnactive.size()$ **do**
    $N \leftarrow VSZ.get(i).vnactive.get(j)$;
    if *(Rule 1.Cdt 1.Cdt 1.1) || (Rule 1.Cdt 2.Cdt 2.1.case 1) || (Rule 1.Cdt 2.Cdt 2.2.case 2) & (Rule 1.!(Cdt 1.Cdt 1.1))* **then**
      $S \leftarrow VSZ.get(i).getSink()$ $S.moveascloseTo(N)$;
    end
    if *((Rule 1.Cdt 2.Cdt 2.1.case 2) & (Rule 1.Cdt 2.!(Cdt 2.2).case 1)) || (Rule 1.Cdt 2.Cdt 2.2.case 1)* **then**
      $M \leftarrow N.getclosestMN()$ $M.moveascloseTo(N)$
    end
  end
end

Algorithm 2: Resizing of zones in R [38]

Require: Set of active zones.
Ensure: Cover the possible largest zones in R.
for $i = 0$ *to* NbZ **do**
  $Z \leftarrow VZ.get(i)$;
  if *Rule 2.Cdt 1* **then**
    $Z1 \leftarrow findAppropriateNeigh()$;
  end
  if $Z \neq null\ AND\ Z1 \neq null$ **then**
    $Z.Ag.deactivate()$;
    for $j = 0$ *to* $Z.Ag.NbNAct$ **do**
      $Z1.Ag.VN.add(Z.Ag.VNAct.get(j))$;
      for $k = 0$ *to* $Z.NbSink$ **do**
        $S \leftarrow Z.Ag.VSZ.get(k).getSink()$;
        if $S.state = 1$ **then**
          $Z1.Ag.VSZ.add(Z.Ag.VSZ.get(k))$;
        end
      end
    end
    $findNeigh(Z1)$;
  end
end

Algorithm 3: Detect the malfunctioning zone agents by Ag_{Ctrl}

Require: Set of active zone agents.
Ensure: Detect the malfunctioning zone agents.
$sendTstPck \leftarrow false$; $VAgTstPck \leftarrow null$;
$AllAgVerif \leftarrow false$; $startTime \leftarrow currentTime()$;
while $AllAgVerif = false$ **do**
  $endTime \leftarrow currentTime()$; $waitTime \leftarrow endTime - startTime$;
  for $i = 0$ *to* $VZ.size()$ **do**
    if $waitTime > dl1(VZ.get(i).Ag, controller)$ **then**
      $VAgTstPck.add(VZ.get(i).Ag)$;
      $sendTstPckTo(VZ.get(i).Ag)$;
      $endTime \leftarrow currentTime()$;
      $waitTime \leftarrow endTime - startTime$;
      if $waitTime > dl2(Ag, controller)$ **then**
        $isolate(Ag)$;
      end
    end
  end
end

6 Case Study

In this section, we present a case study, simulate it using *RWSNSim*, and describe the simulation steps. Then, we evaluate the performance of the proposed methodology.

6.1 Presentation

As a case study, we propose a WSN named *WSN_Forest* and an RWSN named *RWSN_Forest* which are designed to protect the forests against fires to preserve the animals' and humans' lives.

In fact, *WSN_Forest* consists of a base station BS, 4 zones formalized by $S_Z = \{Z_1, Z_2, Z_3, Z_4\}$ where each zone is composed of a gateway denoted by G_k, and 20 sensor nodes defined by $S_N(Z_k) = \{N_{i,k} \mid i \in [1..20] \ \ and \ \ k \in [0..4]\}$. Otherwise, *RWSN_Forest* is composed of a base station BS, a controller agent Ag_{Ctrl}, 4 zones formalized by $S_Z = \{Z_1, Z_2, Z_3, Z_4\}$ where each zone is composed of a zone agent formalized by $Ag_k | k \in [1..4]$, 3 MSNs denoted by $S_{SN}(Z_k) = \{SN_{j,k} \mid j \in [1..3] \mid and \ \ k \in [1..4]\}$, and 20 sensor nodes (12 fixed and 8 mobile ones) defined by $S_N(Z_k) = \{N_{i,k} \ \ i \in [1..20] \ \ and \ \ k \in [1..4]\}$.

Each sensor node $N_{i,k}$ in *WSN_Forest* and *RWSN_Forest* has two sensors: a temperature sensor and a CO_2 sensor.

6.2 Simulation of Case Study

To evaluate the performance of the suggested methodology, we simulate *WSN_Forest* and *RWSN_Forest*. In the following, we show all provided services and simulation steps by *RWSNSim* during the simulation of *RWSN_Forest*.

Step 01 - Choosing the Appropriate Routing Protocol: Before creating a new sensor network with *RWSNSim*, the user has to choose an appropriate routing protocol between LEACH and WBM-TEEN. In this case study, we choose to simulate *RWSN_Forest* using WBM-TEEN protocol.

Step 02 - Creating the *RWSN_Forest* Network: Indeed, *RWSNSim* provides two manners to create a new sensor network and save it in a database. They are explained as follows:

1. *Manual & Regular:* this manner allows the user to enter manually all information related to each entity in the network, except for the identifiers. In this case study, we use this manner to create *RWSN_Forest*.
2. *Automatic & Random:* this manner allows the user to automatically generate automatically all information related to each entity in the target sensor network such as the coordinates, the state, and the batteries' charge of each entity. The system randomly selects the related information according to some inputs such as the total number of zones and the total number of SNs and MSNs in each zone. The user can also modify each generated information in the created database, except for the identifiers of network entities.

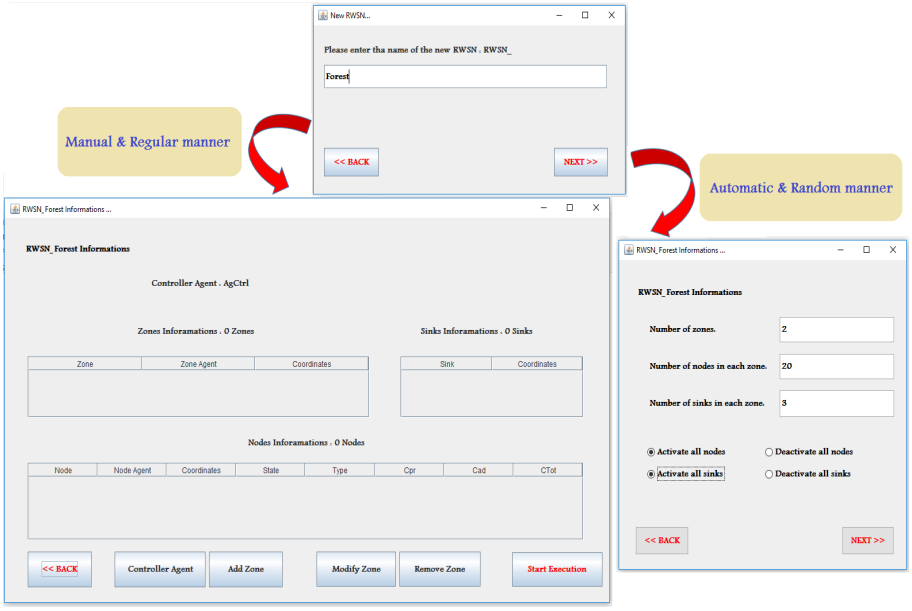

Fig. 11. The two creation manners of *RWSN_Forest*

Figure 11 displays the creation manners in the suggested simulation tool.

In fact, *RWSNSim* stores all information entered by the user or generated automatically in a database after the creation of the target sensor network. Figure 12 shows the *RWSN_Forest* information that is stored in the database used by *RWSNSim*.

Step 03 - Execution of *RWSN_Forest* Network: Before starting the network execution, *RWSNSim* demands a set of parameters and thresholds. After that, two windows appear: the simulation graph window and the execution report window. They are defined as follows:

- **Simulation graph window:** this window plots the different network entities and their distribution in the network. It displays also the communication between them in real-time.
- **Execution report window:** this window provides a comprehensive report of monitoring times and sensing data sent by sensor nodes.

Figure 13 shows the simulation graph and the execution report of *RWSN_Forest* that are captured during the simulation process. .

Step 04 - Notifications & Alerts: The suggested simulation tool provides also many notification and alert windows which appear synchronized with various events that occur during the simulation process. Indeed, there are four types of notification and alert windows: Deactivation notification window, mobility notification window, Resizing notification window, and alert window. Figure 14 displays these windows.

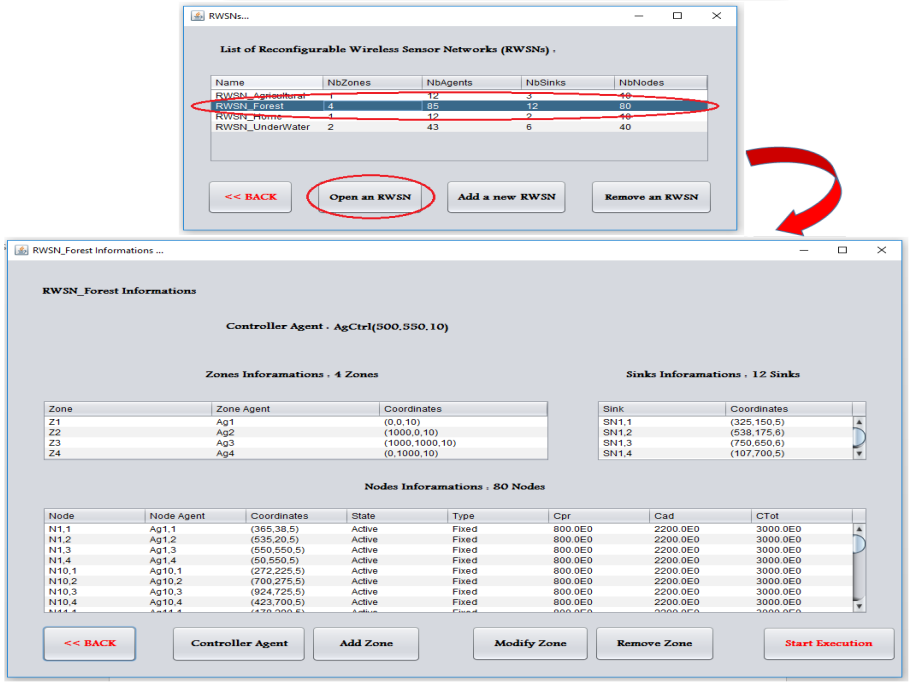

Fig. 12. *RWSN_Forest* informations stored in a database

Step 05 - Draw the Obtained Line Chart of *RWSN_Forest*: After the simulation process of the target sensor network, the suggested simulation tool provides the obtained results in the form of line charts. Figure 15 illustrates the obtained line chart of *RWSN_Forest*.

Through the mentioned provided services and steps of the simulation process, we note that the proposed simulator is not a specific simulator developed to implement a specific methodology, but rather tends to be more generalized. Indeed, it can simulate sensor networks without applying the suggested methodology. It has also the scalability to include new solutions, techniques and routing protocols.

6.3 Evaluation of Performance

In order to figure out the performance of the proposed methodology, we simulate *WSN_Forest* and *RWSN_Forest* using LEACH and WBM-TEEN routing protocols. In the following, we present the obtained results in the worst case, which means during the months when the energy production by each entity E is negligible ($EP_{[\rho]}(E) \approx 0$) and the consumed energy is high ($EC_{[\rho]}(E) > 0$). Figure 16 shows the obtained line charts using *RWSNSim*.

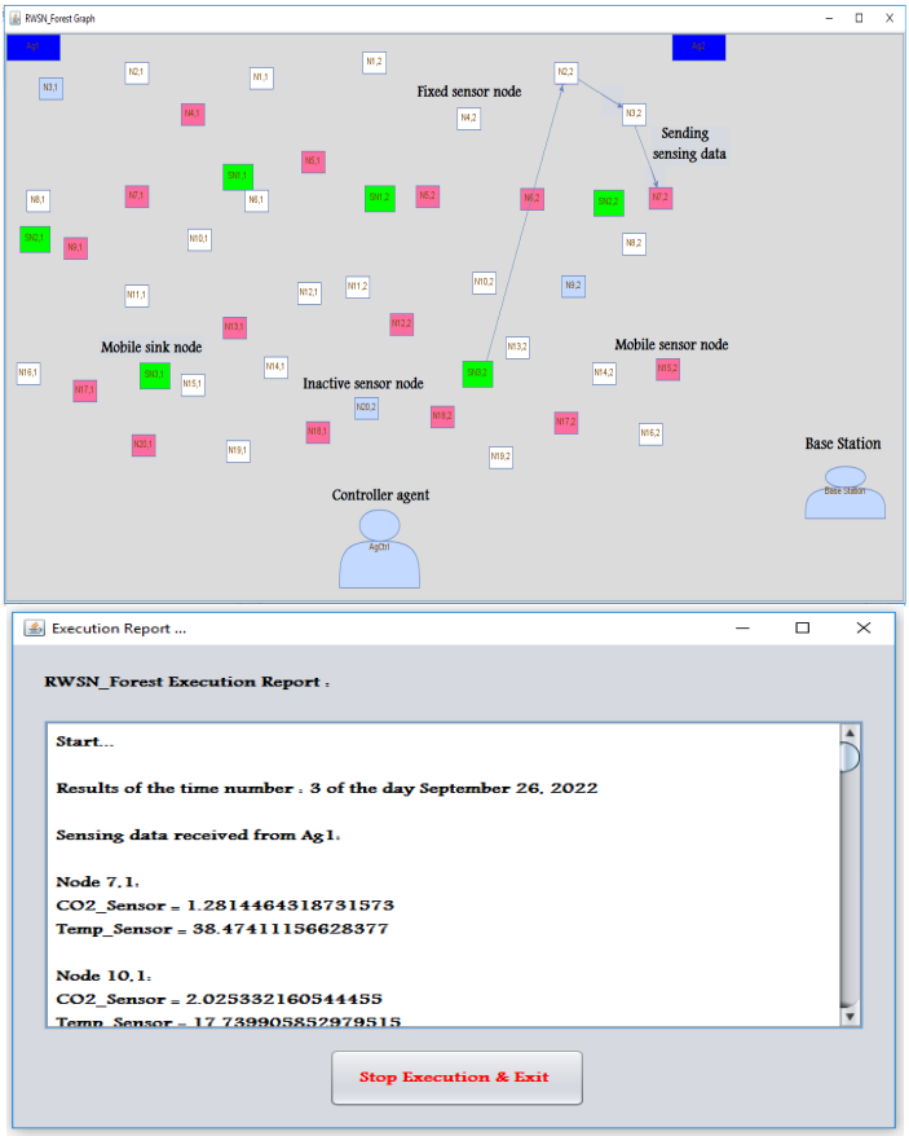

Fig. 13. *RWSN_Forest* simulation graph and execution report

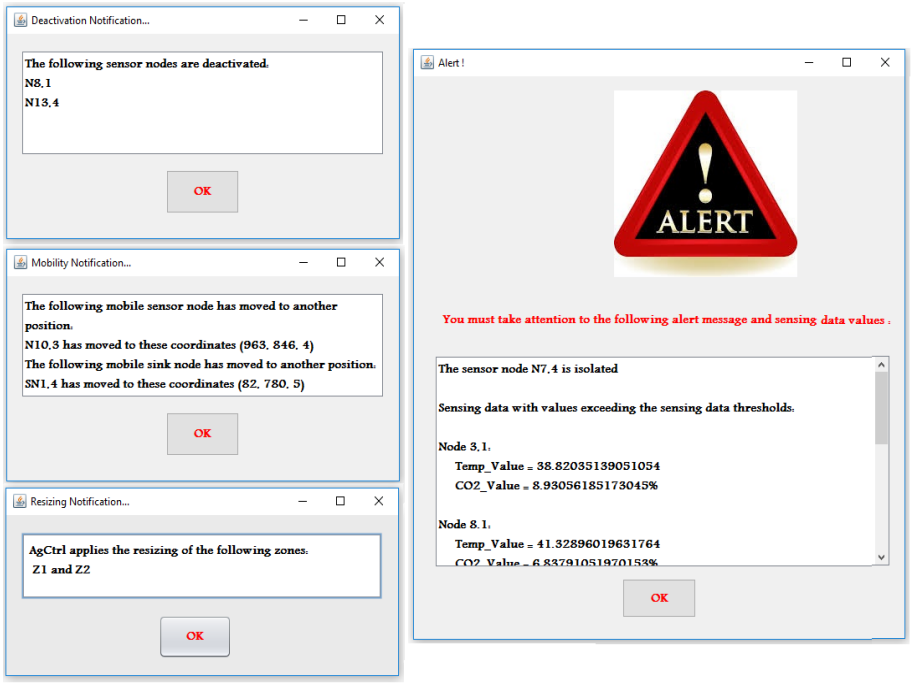

Fig. 14. *RWSN_Forest* notifications and alert windows

Through Figure 16, we remark that:

- By comparing case (b) with case (a) and case (d) with case (c), we remark that the effectiveness of the use of WBM-TEEN protocol achieves a success rate approx 35.82% and 38.36%, respectively.
- By comparing case (c) with case (a) and case (d) with case (b), we remark that the effectiveness of the proposed methodology with the use of the same routing protocol achieves a success rate approx 355,22% and 363.74%, respectively.
- By comparing case (d) with case (a), we remark that the effectiveness of the proposed methodology using WBM-TEEN protocol achieves a success rate approx 529.85% compared to the case when we use LEACH protocol without the proposed methodology.

On the other hand, when a software/hardware failure is committed in an entity E in *WSN_Forest* in case (a) or case (b), the packet dropping problem will occur which may lead to a complete network shutdown. While in case (c) or case (d), the failure will be detected and isolated when it occurs. Thus, eliminating the packet dropping problem and *RWSN_Forest* will still work without problems.

According to the obtained results, we can come up with an important inference with the choose of the appropriate routing protocol. Indeed, it is better to choose LEACH protocol as an appropriate routing protocol in the case that the network needs periodic

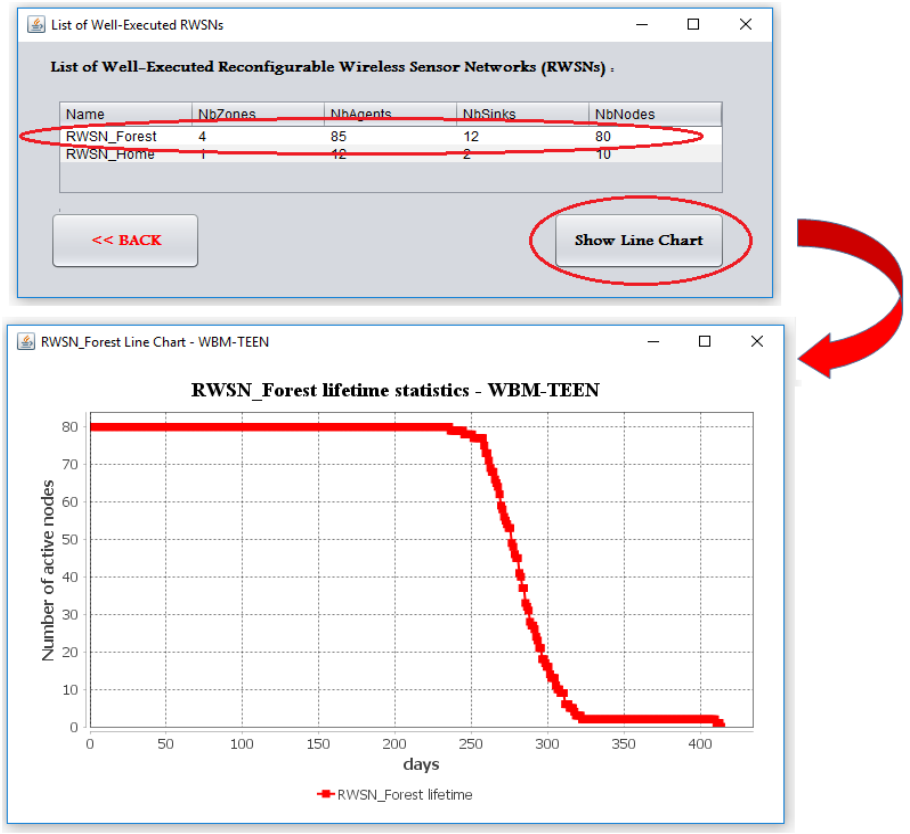

Fig. 15. *RWSN_Forest* obtained line chart

monitoring and the sensor nodes are required to send the sensing data periodically. While it is better to choose WBM-TEEN protocol as an appropriate routing protocol in the case that the network doesn't need periodic monitoring and the sensor nodes are required to send the sensing data only when their values exceed the thresholds.

7 Conclusion

In this work, we described the architecture of sensor networks, formalized the energy model and the energy problem, and explained the hardware & software failures in sensor networks. Moreover, we propose a new methodology to resolve the energy, real-time, and hardware/software failure problems in RWSNs to keep the network alive as long as possible without human intervention. Furthermore, we developed a new simulation tool named *RWSNSim* which is a discrete-event simulator designed for wireless networks. The suggested simulator permits simulating WSNs & RWSNs with and without the proposed methodology using two routing protocol LEACH & WBM-TEEN. Indeed, we simulated an experimentation with *RWSNSim* with and without the proposed

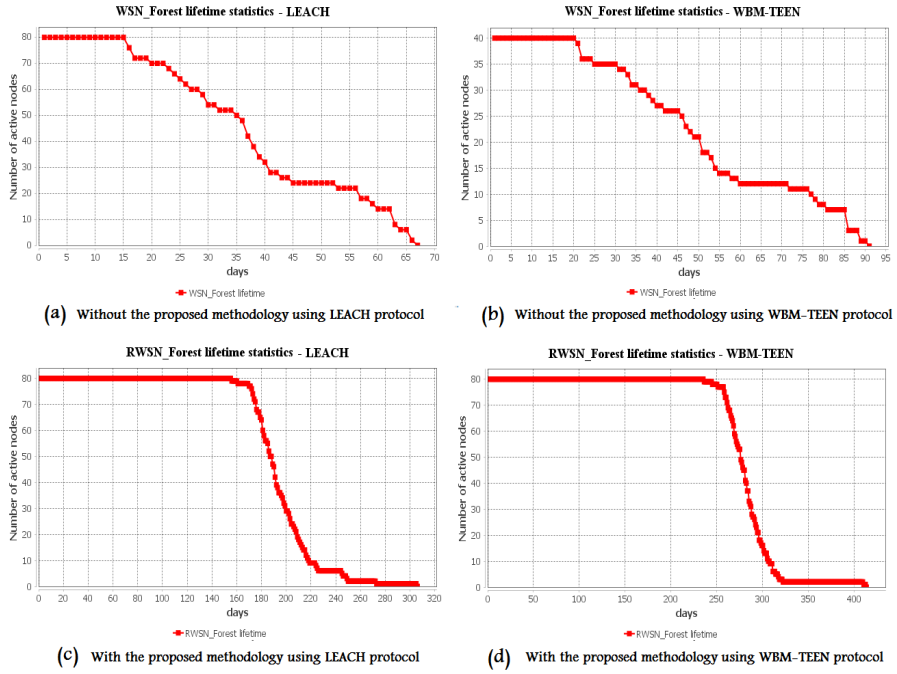

Fig. 16. Obtained line charts

methodology. The results prove the effectiveness of the suggested methodology and the high performance of the developed simulation tool. In the future, we will strive to improve the proposed simulator by adding new routing protocols, new solutions and techniques to resolve more problems in sensor networks, new features, and new types of charts.

References

1. Agrawal, D.P.: Sensor Nodes (SNs), Camera Sensor Nodes (C-SNs), and Remote Sensor Nodes (RSNs), pp. 181–194. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-3038-3_8, ISBN 978-981-10-3037-6
2. Allouch, A., Cheikhrouhou, O., Koubâa, A., Khalgui, M., Abbes, T.: Mavsec: Securing the mavlink protocol for ardupilot/px4 unmanned aerial systems. In: 2019 15th International Wireless Communications Mobile Computing Conference (IWCMC). pp. 621–628 (2019)
3. Ap, P., S., P.: Zone-based sink mobility in wireless sensor networks. Sensor Review **39**(6), 874–880 (09 2019). https://doi.org/10.1108/SR-11-2018-0310
4. Ara, T., Shah, P.: Light weight network security protocol for wireless sensor network (06 2014)
5. Bakhtiar, Q.A., Makki, K., Pissinou, N.: Data reduction in low powered wireless sensor networks. In: Matin, M. (ed.) Wireless Sensor Networks - Technology and Applications, chap. 8. IntechOpen, Rijeka (07 2012). https://doi.org/10.5772/50178, ISBN 978-953-51-0676-0

6. Chao, F., He, Z., Pang, A., Zhou, H., Ge, J., Baños, R.: Path optimization of mobile sink node in wireless sensor network water monitoring system. Complexity **2019**, 1–10 (11 2019). https://doi.org/10.1155/2019/5781620, ISSN 1076-2787
7. Duche, R.N., Sarwade, N.P.: Sensor node failure detection based on round trip delay and paths in wsns. IEEE Sensors Journal **14**(2), 455–464 (2014). https://doi.org/10.1109/JSEN.2013.2284796
8. Elsayed, W., Elhoseny, M., Riad, A.M., Hassanien, A.E.: Autonomic self-healing approach to eliminate hardware faults in wireless sensor networks. In: Hassanien, A.E., Shaalan, K., Gaber, T., Tolba, M.F. (eds.) Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2017. pp. 151–160. Springer International Publishing, Cham (2018)
9. Erman, A.T., Dilo, A., Havinga, P.: A virtual infrastructure based on honeycomb tessellation for data dissemination in multi-sink mobile wireless sensor networks. EURASIP Journal on Wireless Communications and Networking **2012**(1), 17 (2012). https://doi.org/10.1186/1687-1499-2012-17
10. Gasmi, M., Mosbahi, O., Khalgui, M., Gomes, L., Li, Z.: R-node: New pipelined approach for an effective reconfigurable wireless sensor node. IEEE Transactions on Systems, Man, and Cybernetics: Systems **48**(6), 892–905 (12 2016). https://doi.org/10.1109/TSMC.2016.2625817
11. Ghribi, I., Abdallah, R.B., Khalgui, M., Li, Z., Alnowibet, K.A., Platzner, M.: R-codesign: Codesign methodology for real-time reconfigurable embedded systems under energy constraints. IEEE Access **6**, 14078–14092 (2018)
12. GloMoSim: GloMoSim Simulator Projects : Online Network Simulator : Network Simulation Tools. https://networksimulationtools.com/glomosim-simulator-projects/ (2020), [Online; accessed 27-March-2022]
13. Grichi, H., Mosbahi, O., Khalgui, M.: A development tool chain for reconfigurable wsns. In: Fujita, H., Papadopoulos, G.A. (eds.) New Trends in Software Methodologies, Tools and Techniques - Proceedings of the Fifteenth SoMeT_16, Larnaca, Cyprus, 12-14 September 2016. Frontiers in Artificial Intelligence and Applications, vol. 286, pp. 101–114. IOS Press (2016). https://doi.org/10.3233/978-1-61499-674-3-101
14. Grichi, H., Mosbahi, O., Khalgui, M., Li, Z.: Rwin: New methodology for the development of reconfigurable wsn. IEEE Transactions on Automation Science and Engineering **14**(1), 109–125 (01 2017). https://doi.org/10.1109/TASE.2016.2608918
15. Grichi, H., Mosbahi, O., Khalgui, M., Li, Z.: New power-oriented methodology for dynamic resizing and mobility of reconfigurable wireless sensor networks. IEEE Transactions on Systems, Man, and Cybernetics: Systems **48**(7), 1120–1130 (2018). https://doi.org/10.1109/TSMC.2016.2645401
16. Hafidi, Y., Kahloul, L., Khalgui, M., Li, Z., Alnowibet, K., Qu, T.: On methodology for the verification of reconfigurable timed net condition/event systems. IEEE Transactions on Systems, Man, and Cybernetics: Systems **50**(10), 3577–3591 (2020)
17. Housseyni, W., Mosbahi, O., Khalgui, M., Li, Z., Yin, L., Chetto, M.: Multiagent architecture for distributed adaptive scheduling of reconfigurable real-time tasks with energy harvesting constraints. IEEE Access **6**, 2068–2084 (2018). https://doi.org/10.1109/ACCESS.2017.2781459
18. Ji, S., Beyah, R., Cai, Z.: Snapshot and continuous data collection in probabilistic wireless sensor networks. IEEE Transactions on Mobile Computing **13**(3), 626–637 (03 2014). https://doi.org/10.1109/TMC.2013.30, ISSN 1536-1233
19. JSim: GitHub - nmcl/JavaSim: JavaSim simulation classes and examples. https://github.com/nmcl/JavaSim (2022), [Online; accessed 27-March-2022]

20. Khediri, S.E., Nasri, N., Khan, R.U., Kachouri, A.: An improved energy efficient clustering protocol for increasing the life time of wireless sensor networks. Wirel. Pers. Commun. **116**(1), 539–558 (2021)
21. Khriji, S., Houssaini, D., Kammoun, I., Kanoun, O.: Energy-efficient techniques in wireless sensor networks: Technology, Components and System Design, pp. 287–304 (11 2018). https://doi.org/10.1515/9783110445053-017, ISBN 978-311-044-505-3
22. Khriji, S., Houssaini, D.E., Kammoun, I., Kanoun, O.: Energy-efficient techniques in wireless sensor networks: Technology, Components and System Design, p. 287–304. De Gruyter (11 2018). https://doi.org/10.1515/9783110445053-017, ISBN 978-311-044-505-3, http://dx.doi.org/10.1515/9783110445053-017
23. Moridi, E., Haghparast, M., Hosseinzadeh, M., Jafarali Jassbi, S.: Novel fault-tolerant clustering-based multipath algorithm (ftcm) for wireless sensor networks. Telecommunication Systems **74**(4), 411–424 (08 2020). https://doi.org/10.1007/s11235-020-00663-z
24. Moussa, N., Alaoui, A.E.B.E., Chaudet, C.: A novel approach of WSN routing protocols comparison for forest fire detection. Wirel. Networks **26**(3), 1857–1867 (2020)
25. Moussa, N., El Belrhiti El Alaoui, A.: A cluster-based fault-tolerant routing protocol for wireless sensor networks. International Journal of Communication Systems **32**(16), e4131 (2019). https://doi.org/10.1002/dac.4131, https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.4131, e4131 dac.4131
26. Nakas, C., Kandris, D., Visvardis, G.: Energy efficient routing in wireless sensor networks: A comprehensive survey. Algorithms **13**(3) (2020). https://doi.org/10.3390/a13030072, ISSN 1999-4893, https://www.mdpi.com/1999-4893/13/3/72
27. Nayyar, A., Singh, R.: A comprehensive review of simulation tools for wireless sensor networks (wsns). Wireless Networking and Communications **5**(1), 19–47 (01 2015). https://doi.org/10.5923/j.jwnc.20150501.03
28. Nguyen, L., Nguyen, H.T.: Mobility based network lifetime in wireless sensor networks: A review. Computer Networks **174**, 107236 (06 2020). https://doi.org/10.1016/j.comnet.2020.107236, ISSN 1389-1286, https://www.sciencedirect.com/science/article/pii/S1389128619315865
29. NS3: ns-3 : a discrete-event network simulator for internet systems. https://www.nsnam.org/ (2022), [Online; accessed 26-March-2022]
30. OMNeTPP: OMNeT++ Discrete Event Simulator. https://omnetpp.org/ (2019), [Online; accessed 27-March-2022]
31. Peijun, Z., Feng, R.: An energy efficient multiple mobile sinks based routing algorithm for wireless sensor networks. IOP Conference Series: Materials Science and Engineering **323**, 012029 (03 2018). https://doi.org/10.1088/1757-899x/323/1/012029
32. Priya, M., C, S., k.a, S.: Proactive fault tolerant routing protocol for mobile wsn **9**, 1892–1896 (11 2019). https://doi.org/10.35940/ijitee.L3656.119119
33. Raj, S.N., Bhattacharyya, D., Midhunchakkaravarthy, D., Kim, T.: Multi-hop in clustering with mobility protocol to save the energy utilization in wireless sensor networks. Wireless Personal Communications **117**(4), 3381–3395 (01 2021). https://doi.org/10.1007/s11277-021-08078-y
34. Rajan, C., Geetha, K., Priya, C.R., Geetha, S., Manikandan, A.: A simple analysis on novel based open source network simulation tools for mobile ad hoc networks. International Journal of Advanced Research in Computer Science and Software Engineering **5**(3), 716–721 (3 2015)
35. Ramasamy, V.: Mobile wireless sensor networks: An overview. In: Sallis, P. (ed.) Wireless Sensor Networks - Insights and Innovations, chap. 1, pp. 1–12. IntechOpen, Rijeka (10 2017). https://doi.org/10.5772/intechopen.70592, ISBN 978-953-51-3562-3; 978-953-51-3561-6

36. Robinson, H., Julie, G., Balaji, S.: Bandwidth and delay aware routing protocol with scheduling algorithm for multi hop mobile ad hoc networks. International Journal of Computer, Electrical, Automation, Control and Information Engineering **10**(8), 1512–1521 (02 2017)
37. Rouainia, H., Grichi, H., Kahloul, L., Khalgui, M.: 3d mobility, resizing and mobile sink nodes in reconfigurable wireless sensor networks based on multi-agent architecture under energy harvesting constraints. In: Proceedings of the 15th International Conference on Software Technologies - ICSOFT,. pp. 394–406. INSTICC, SciTePress (01 2020). https://doi.org/10.5220/0009971503940406, ISBN 978-989-758-443-5; ISSN 2184-2833
38. Rouainia, H., Grichi, H., Kahloul, L., Khalgui, M.: New energy efficient and fault tolerant methodology based on a multi-agent architecture in reconfigurable wireless sensor networks. In: Kaindl, H., Mannion, M., Maciaszek, L.A. (eds.) Proceedings of the 17th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2022, Online Streaming, April 25-26, 2022. pp. 405–416. SCITEPRESS (2022). https://doi.org/10.5220/0011061300003176
39. Rouainia, H., Grichi, H., Kahloul, L., Khalgui, M.: Reconfigurable wireless sensor networks simulator (rwsnsim): A new discrete-event simulator. In: Fill, H., van Sinderen, M., Maciaszek, L.A. (eds.) Proceedings of the 17th International Conference on Software Technologies, Lisbon, Portugal, July 11-13, 2022. pp. 349–361. SCITEPRESS (2022). https://doi.org/10.5220/0011318300003266, ISBN 978-989-758-588-3; ISSN 2184-2833
40. RWiN: RWiN-Project: New Solutions for Reconfigurable Wireless Sensor Networks. https://lisi-lab.wix.com/rwinproject (2016), [Online; accessed 26-March-2022]
41. RWSNSim: RWSNSim: Reconfigurable Wireless Sensor Networks Simulator. https://hanenerouainia.wixsite.com/rwsnsim (2022), [Online; accessed 20-April-2022]
42. Salem, M.O.B.: BROMETH: Methodology to Develop Safe Reconfigurable Medical Robotic Systems: Application on Pediatric Supracondylar Humeral Fracture. Ph.D. thesis, Saarland University, Saarbrücken, Germany (2017)
43. Shalini, V.B., Vasudevan, V.: e-nl beenish: extended-network lifetime balanced energy efficient network integrated super heterogeneous protocol for a wireless sensor network. Int. J. Comput. Aided Eng. Technol. **15**(2-3), 317–327 (2021)
44. Thomas, S., Mathew, T.: Intelligent path discovery for a mobile sink in wireless sensor network. Procedia Computer Science **143**, 749 – 756 (2018). https://doi.org/10.1016/j.procs.2018.10.430, ISSN 1877-0509, http://www.sciencedirect.com/science/article/pii/S1877050918321288, 8th International Conference on Advances in Computing and Communications (ICACC-2018)
45. Tzounis, A., Katsoulas, N., Bartzanas, T., Kittas, C.: Internet of things in agriculture, recent advances and future challenges. Biosystems Engineering **164**, 31–48 (12 2017). https://doi.org/10.1016/j.biosystemseng.2017.09.007
46. Verma, S., Sood, N., Sharma, A.: A novelistic approach for energy efficient routing using single and multiple data sinks in heterogeneous wireless sensor network. Peer-to-Peer Networking and Applications **12**(5), 1110–1136 (09 2019). https://doi.org/10.1007/s12083-019-00777-5
47. Vijayalakshmi, S., Muruganand, S.: Wireless Sensor Network: Architecture - Applications - Advancements (05 2018)
48. Wang, J., Gao, Y., Liu, W., Kumar, A., Kim, H.J.: Energy efficient routing algorithm with mobile sink support for wireless sensor networks. Sensors **19**(7), 1494 (03 2019). https://doi.org/10.3390/s19071494
49. Zagrouba, R., Kardi, A.: Comparative study of energy efficient routing techniques in wireless sensor networks. Information **12**(1), 42 (01 2021). https://doi.org/10.3390/info12010042
50. Zidi, S., Moulahi, T., Alaya, B.: Fault detection in wireless sensor networks through svm classifier. IEEE Sensors Journal **18**(1), 340–347 (2018). https://doi.org/10.1109/JSEN.2017.2771226