Article

Energy and Memory Efficient Data Loss Prevention in Wireless Sensor Networks

Pooya Hejazi [2,†,‡], and Gianluigi Ferrari [1,*]

[1] Affiliation 1; Internet of Things (IOT) Lab, Department of Engineering and Architecture, University of Parma, Parco Area delle Scienze, 181/A, 43124 Parma PR, Italy
[2] Affiliation 2; Department of Computer Engineering and Infomation Technology, Eqbal Lahoori Institute of Higher Education, Mashhad, KHR, Iran.
* Correspondence: gianluigi.ferrari@unipr.it ; Tel.: +39-0521-906513
† Current address: Internet of Things (IOT) Lab, Department of Engineering and Architecture, University of Parma, Parco Area delle Scienze, 181/A, 43124 Parma PR, Italy
‡ These authors contributed equally to this work.

Abstract: Load balancing, energy efficiency and fault tolerance are among the most important data dissemination issues in Wireless Sensor Networks (WSNs). In order to successfully cope with the mentioned issues, two main approaches (namely, Data-centric Storage and Distributed Data Storage) have been proposed in the literature. Both approaches suffer from data loss due to memory and/or energy depletion in the storage nodes. Even though several techniques have been proposed so far to overcome the mentioned problems, the proposed solutions typically focus on one issue at a time. In this paper, we integrate the Data-centric Storage (DCS) features into Distributed Data Storage (DDS) mechanisms and present a novel approach, denoted as Collaborative Memory and Energy Management (CoMEM), to overcome both problems and bring memory and energy efficiency to the data loss mechanism of WSNs. We also propose analytical and simulation frameworks for performance evaluation. Our results show that the proposed method outperforms existing approaches in various WSN scenarios.

Keywords: storage and retrieval processes; load-balancing; fault tolerance; energy efficiency; memory efficiency; data loss

1. Introduction

A WSN is a collection of small sensing devices with limited bandwidth, power, and computational capabilities. The main goal of a WSN is to gather information from specific environment, for applications such as remote monitoring and target tracking. The design of a WSN depends significantly on the application and must consider factors such as the deployment environment, the application's design objectives, the maximum cost, the available hardware, and system constraints [1]. Energy efficiency, fault tolerance, and load balancing are among the most challenging issues of a WSN. Therefore, many approaches have been proposed in the literature trying to handle the mentioned issues in order to make a WSN reliable and scalable. In particular, data storage plays a key role in making a WSN effective. Data storage approaches can be categorized into two groups: Distributed Data Storage (DDS) and Data-centric Storage (DCS).

To cope with energy efficiency, DDS approaches concentrate on local data storage in sensing nodes. This makes the data storage process very energy efficient. However, in order to collect the specific information stored in the network, all sensors have to be queried by means of a flooding mechanism: this makes the data retrieval process incurs a high amount of traffic and decreases the network lifetime. On the other hand, DCS approaches focus on data-centric mechanisms for storage and retrieval by means of Geographic Hash Tables (GHTs) [2] and geometric routing algorithms such as Greedy Perimeter Stateless Routing (GPSR) [3]. Although DCS is more energy consuming than DDS in the storage process, its query process is based on directed dissemination of queries to a specific node, denoted as *storage node*, so that it reduces the query traffic and increases the efficiency of the retrieval process.

Data replication over multiple storage nodes is the mechanism which both groups of data storage techniques use to handle fault tolerance and load-balancing. Fault tolerance in the WSN is obtained by preventing data loss caused by energy or memory depletion in the storage node. The approaches in [4–10] focus on data loss prevention due to energy shortages, whereas the approaches in [11–13] concentrate on efficient memory usage of nodes in the WSN. The mechanism with which a replica is elected in DDS is based on local broadcasts between the neighbors of the storage node. This fully distributed mechanism generates a high amount of traffic and decreases the network lifetime.

In this paper, we integrate DCS replication features into DDS to cover both memory and energy efficiency in data loss prevention mechanisms for WSNs. We present a novel mechanism denoted as Collaborative Memory and Energy Management (CoMEM). We divide the WSN into multiple zones: in each zone we include a monitor node, proposed in [8], to act as a gateway for both storage and retrieval processes. Therefore, if an event is sensed throughout the zone, the collected information will be forwarded toward the monitor, which stores the summary of the received data locally and chooses the appropriate node for storing the details. On the other hand, in the retrieval process, all queries are routed toward the monitor, which decides whether to answer the query directly (summarized retrieval) or to redirect the query to a specific storage node.

In order to bring self-organization, load balancing, and fault tolerance, two relevant parameters are considered in each zone. The first parameter is related to the percentage of memory and energy availability at the storage nodes. If this parameter falls below a properly set threshold, the monitor will replicate the data over the most memory and energy available node within the zone and will balance the storage traffic load between the previous storage node and its new replica. The second parameter is related to the percentage of energy availability in the monitor. As it acts as a gateway for both storage and retrieval processes, it will deplete its energy quickly. Whenever the percentage of the energy availability of the monitor falls below the properly determined threshold, the most energy-available node is chosen by the current storage node to be the new monitor. As a result, the mechanism collaboratively chooses new replicas and monitors and brings self-organization, fault tolerance, and load balancing to the WSN.

The rest of the paper is organized as follows. Section 2 is devoted to related works on memory shortage problems leading to CoMEM in WSNs. Section 3 proposes our appoarch to integrate DCS features into DDS mechanisms, Section 4 describes the simulation framework used for performance evaluation, Section 5 is dedicated to simulation results. Finally, Section 6 concludes the paper.

2. Related Works

In [14], fault tolerance WSN applications are categorized into five groups: (i)node placement, (ii)topology control, (iii)target and event detection, (iv)data gathering and aggregation, and (v)sensor monitoring and surveillance. In this section, we will briefly take a look at works previously proposed in the area of memory efficient data gathering in WSNs. Basic DCS and DDS methods are not designed either for sensornets with highly mobile or unreliable nodes. Thus, enhanced techniques must be designed so that data dissemination mechanisms are robust against data losses due to both energy and memory depletion occurrences in nodes. As several researches are dedicated to energy efficiency in WSN; our effort mainly concentrates on memory efficient data dissemination mechanisms.

In [11], a Low Complexity Distributed Data Replication (LCDDR) mechanism is proposed to prevent data loss due to memory shortages by means of replicating the data over the most memory available node denoted as “donor node.” This brings fault tolerance and reliability to the data storage mechanism. However, the mentioned method has some challenging issues. First, according to the results in [11], the query process is of no concern in the proposed mechanism. Second, the local broadcast mechanism for donor node election is not energy efficient for high scale data dissemination in WSNs. Third, due to the fully distributed donor node election mechanism of LCDDR, a hijacking node can declare itself as the most memory available node and become the donor node in the election process, thus easily sniffing data. Finally, if no donor node can be chosen, the data loss will happen due to the lack of a widespread election mechanism.

In [12], a hybrid routing algorithm, denoted as Bloom Filter based Routing Protocol (BFRP), is proposed: it relies on hierarchical (cluster-based) routing and bloom filter data aggregation. This method decreases the amount of memory usage for the routing table at each node by means of bloom filtering and cluster head data aggregation. However, cluster head election is handled by local broadcasts between the neighbors of the current cluster head. Each node broadcasts a parameter, denoted as "coverage-aware cost" and calculated according to its remaining energy, to its neighbors. After the first broadcast, the node waits for the specific amount of time determined by the coverage-aware cost and, if there is no announcement with a lower cost, it will declare itself as a cluster head to the neighbors by means of another broadcast. This generates a high amount of traffic for cluster head election. Moreover, the elected node is the most energy available one and memory usage is of no concern.

Another method, which relies on data compression and, thus, brings memory efficiency, is presented in [13]. In this case, the memory usage of nodes and the wireless bandwidth consumption is affected by a co-design memory mechanism for low-latency in-place lightweight compression. However, as the compression happens on relevant memory pages, the information can only be accessed by multiple packet payload decompression. Therefore, data aggregation on cluster head will bring a high amount of energy consumption. This tends to deplete the energy of the cluster head. Moreover, no election mechanism for a depleted cluster head is proposed.

Memory-based Message Efficient Clustering (MMEC) is presented in [15] to enhance energy efficiency of nodes by reducing the propagation of duplicated messages. Moreover, a heuristic protocol is proposed to predefine the distribution of the sensors over the specific sensed area. This protocol increases the energy saving efficiency in the given network. However, no data loss prevention mechanism is proposed and practical issues, such as random distribution of the sensors, are not considered.

A new transport layer protocol, denoted as Reliable Transport with Memory Consideration (RTMC), is presented in [16]: hop-by-hop retransmission guarantees reception of specific data by the sink. It also prevents the change of transmission rate for congestion control to safeguard the control packets. Furthermore, information on the memory status of each sensor is included in the packet header for memory overflow prevention. However, inserting sensor memory information in each packet continuously causes unnecessary redundant data transmissions. Therefore, this protocol is not energy efficient.

Tiny Distributed Shared Memory (TinyDSM) [17] is a reliable DDS mechanism which tries to replicate the data in some nodes along the path between the sink and the source. According to this method, each node along the query propagation path, which contains the replicated data, will answer the query directly. Therefore, the amount of query propagation traffic is load-balanced over multiple replicas of the storage node. However, as replicas are periodically updated by the source, the latter becomes the bottleneck and depletes its energy fast. Also, TinyDSM does not consider energy and memory constraints in its replica selection mechanism.

A fault recovery mechanism in single-hop small WSNs is proposed in [18], as a mechanism for data loss prevention. In this method, the sensed data is shared between all nodes when a storage node is unavailable. Then, by the time the storage node becomes accessible or some sort of election mechanism chooses a new one, the nodes empty their memories, transferring the sensed data to the storage node. Therefore, whenever the frequency of sensed data is high or the WSN has a high node spatial density, the new storage node has to cope with a high amount of storage-related data traffic and depletes its memory and energy fast. Therefore, this method is not memory and energy efficient in highly dense WSNs.

Table 1. Performance characterization of methods.

	LCDDR [11]	BFRP [12]	MMEC [15]	RTMC [16]	TinyDSM [17]	single-hop [18]
Energy Efficiency	No	No	Yes	No	No	No
Memory Efficiency	Yes	Yes	Not mentioned	No	No	No
Load Balancing	Yes	No	No	No	Yes	No
Fault Tolerance	Yes	No	No	Yes	No	Yes
Resolving Single Point Of Failure	Yes	No	No	Yes	Yes	Yes
Complexity Of Method	High	High	low	High	High	low
Performance In Storage Process	Yes	Yes	Yes	No	No	No
Performance In Query Process	No	No	Not mentioned	No	Yes	Yes
Security	No	No	No	No	No	No

In Table 1, a summary of the proposed mechanisms is shwon, considering key performance indicators in WSN scenarios.

3. Collaborative Memory and Energy Management (CoMEM)

Data storage and retrieval processes are interesting design aspects, as they are directly related to a WSN lifetime and to its topological structure. As anticipated in Section 1, DDS and DCS are the main approaches proposed in literature for data storage and retrieval with energy efficiency and scalability.

- In DDS, the source node, upon sensing a specific data, stores it locally without any communication requirement. Therefore, this approach is the most energy efficient one in the storage process. Moreover, in order to collect the sensed data in the retrieval phase, all nodes within the network have to be queried and this typically involves the use of flooding mechanisms. Therefore, the retrieval process involves a high amount of traffic and decreases the network lifetime. Figure 1 shows the retrieval process in DDS-based methods. As illustrated in the figure, the Personal Digital Assistant (PDA) node sends the query for specific data to all nodes. The nodes, which contain the specific data of interest, will answer the PDA. For example, this data could refer to specific targets. The target node specified in the figure can also be mobile and detected by multiple sensors simultaneously.
- In DCS-based methods, the sensed data is stored in a specific location determined by a GHT mechanism [2], which maps the specific event-type to a specific location. In order to route the sensed data, geometric routing algorithms, such as Greedy Perimeter Stateless Routing (GPSR) [3], are used to deliver the data to the nearest node to the mapped location. This node is denoted as *storage node*. Although this approach relatively increases the storage process traffic, the retrieval process is very efficient as all queries are forwarded to the single storage node using the GHT mapped location. This avoids flooding mechanisms and brings energy efficiency and scalability to the network. Figure 2 shows the storage and retrieval processes in DCS-based methods. As shown in the figure, sensors will forward their sensed data to the storage node. Therefore, the PDA only sends queries to this specific node instead of flooding all nodes with queries.

Both basic versions of DDS and DCS suffer from data loss due to node mobility and node failure. To overcome these limitations, several approaches have been proposed in the literature. All of them are based on replicating the data, i.e., creating replicas over multiple storage nodes. In DDS-based approaches, due to a fully distributed storage mechanism, the most memory or energy available node is elected as a replica based on the local broadcasts between the neighbors of the storage node [11], [12]. This mechanism has the following challenging issues. First, a local broadcast mechanism for replica election is not energy-efficient for high densed data dissemination in WSNs. Second, a hijacking node can declare itself as the most suitable node to store the data so that the data can easily be sniffed.

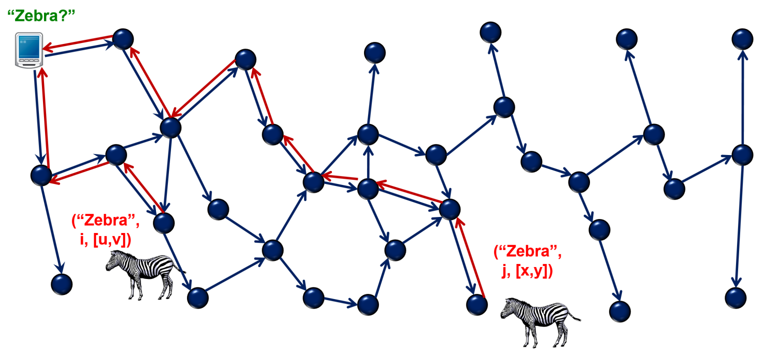

Figure 1. DDS retrieval process.

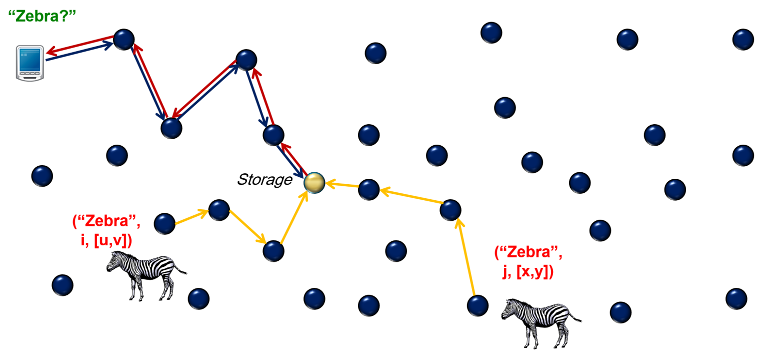

Figure 2. DCS storage and retrieval processes

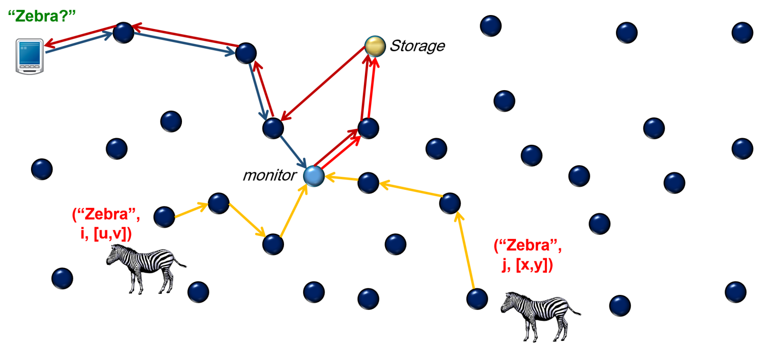

Figure 3. R-DCS storage and retrieval processes

Finally, if no storage node for a replica can be chosen, data loss will be unavoidable. This is due to the lack of a widespread election mechanism.

On the other hand, in DCS-based approaches, the replica is selected by the centralized GHT mechanism: the nearest node to the mapped location, determined by a hash function, will become the replica. Therefore, the energy or memory availability of the new elected replica is of no concern.

According to the method proposed in [8], the entire network is divided into partitions called "zones." In each zone, in addition to the storage node for each event-type, a new node called "monitor" is introduced to act as a gateway for both storage and retrieval processes. Whenever a data is sensed by a source node in the zone, it is forwarded to the monitor by means of a GHT hash function. The monitor then stores the summary of received data locally and forwards its details to the replica specified for the event-type of the received data. In the retrieval process, all queries are forwarded to the monitor node. If the queries are related to locally stored data, the monitor will answer directly; otherwise, it will forward the query to the appropriate replica. However, due to the role of the monitor, all storage and retrieval traffic passes through the monitor and its neighbors. Therefore, they have to support a high amount of traffic, especially when the occurrence frequency of sensed events and the sensor spatial density are high. This depletes the energies of the monitor's neighbors and soon the monitor is left with no neighbor. The neighborless monitor is not accessible and the storage and query traffic will be interrupted. Figure 3 shows the storage and retrieval processes in R-DCS.

We integrate the DCS data storage mechanism into the DDS one and propose a novel scheme, denoted as CoMEM, which relies on the ideas presented in [11], [8] and [9]. According to CoMEM approach, we divide the network into zones with similar dimensions. The number of zones is assumed to be a power of 2 and depends on the network size. In some approaches, e.g., in [9], the number of zone changes on the basis of the occurrence frequency of sensed events. Here, we assume that the number of zones is constant. First, the monitor and storage nodes are determined by GHT within each zone. After this centralized election, a self-organized mechanism for the election of the monitor and replicas is considered. Each node within a zone informs the monitor of its remaining memory and energy. This can be carried out during the propagation of sensed data to the monitor or through a dedicated packet forward. We denote $\zeta_e \in [0,1]$ as the fraction of the remaining energy of the node and $\zeta_m \in [0,1]$ as the fraction of the remaining free memory; These coefficients can thus be expressed as follows:

$$\zeta_e = \frac{r_e}{i_e} \tag{1}$$

$$\zeta_m = \frac{r_m}{i_m} \tag{2}$$

where: i_e is the initial energy (dimension: [J]) and r_e is the remaining energy (dimension: [J]) of each node; i_m is the initial available memory (dimension: [Bytes]) and r_m is the remaining memory (dimension: [Bytes]) of each node. The mentioned coefficients are calculated by each node and sent to the monitor. The monitor stores the received data in its local coefficient table and calculates the "election coefficient" ($\zeta_{replica}$) which is the following heuristic coefficient simultaneously taking into account energy and memory resources:

$$\zeta_{replica} = c_1 \times \zeta_m + c_2 \times \zeta_e \tag{3}$$

$$c_1 + c_2 = 1 \tag{4}$$

where c_1 and c_2 are the relative weights for the energy and memory coefficients. The monitor also computes the following variation coefficient:

$$\zeta_{var(replica)} = (max)\zeta_{replica} - \zeta_{replica} \tag{5}$$

where $\zeta_{replica}$ is the election coefficient of the storage node and $(max)\zeta_{replica}$ is the maximum value of all election coefficients reported by the nodes. If $\zeta_{var(replica)}$ overcomes a pre-defined threshold, i.e., the average value of all election coefficients, the following election mechanism starts:

```
import E(n) #array of energy coefficients
import M(n) #array of memory coefficients

def cover(n) =  ones(n)

    #evaluating the array of covers
    for i in range(1,n)
         for j in range(1,n)
             if  cover(j)<>0 and E(i)<E(j) and M(i)<M(j) and i<>j
              #node j covers node i
              cover(i) = 0
              cover(j) = cover(j) + 1

     #finding the appropriate node for replication
     elected = 1;
     for k in range(2,n)
      if cover(k) > cover(elected)
        elected = k
      else if cover(k)=cover(elected) and E(k)>E(elected)
        elected = k
      else if cover(k)=cover(elected) and E(k)=E(elected)
          and M(k)>M(elected)
        elected = k

     return elected
```

Listing 1: Election algorithm.

If a node has lower energy and/or memory than at least another node in the zone, then the former will be covered by latter and its coverage number (defined in the election algorithm) will be set to zero. As a result, nodes with larger than zero coverage numbers are more likely to become replicas. With this algorithm, local broadcasts between the neighbors of the storage nodes are avoided and also the new replicas are not limited to the neighbors of the storage nodes, which also tolerate a high amount of traffic.

This mechanism has one drawback, normally related to the single point of failure due to the presence of the monitor. Since all the interactions for storage, query and election processes are handled by the monitor, the monitor manages a high amount of traffic and, thus, depletes its energy fast. We then propose an efficient election mechanism for the monitor. This election is performed by the storage

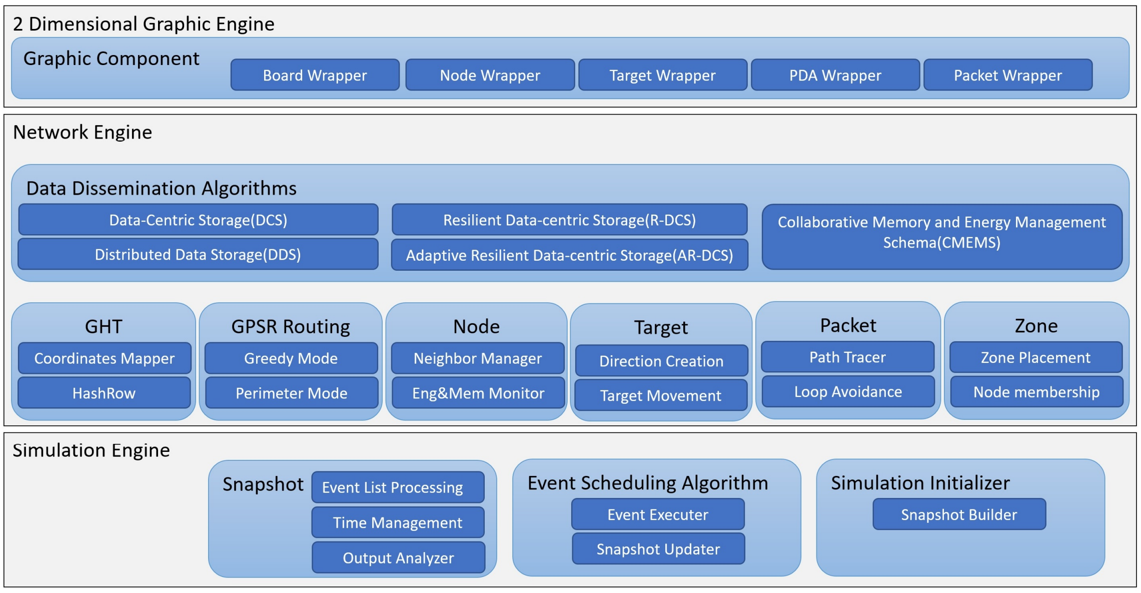

Figure 4. Developed DES structure.

nodes. The collected $\imath_e$ of all nodes are periodically sent to the storage node by the monitor. The storage node then calculates the variation of ζ_e as follows:

$$\zeta_{var(monitor)} = (max)\zeta_e - \zeta_e \tag{6}$$

where ζ_e is energy coefficient of the monitor and $(max)\zeta_e$ is the maximum value of energy coefficients reported by nodes within a zone. If $\zeta_{var(monitor)}$ overcomes a threshold, i.e., the average value of ζ_e of all nodes, the $(max)\zeta_e$ node becomes the new monitor by the storage node announcement throughout the zone. Then, the old monitor transfers its data to the new one. Although this mechanism performs a single broadcast for monitor election in the zone, it can be ignored as it happens whenever the monitor needs to be changed. Therefore, this is more energy efficient than the one presented in DDS approaches. Moreover, as the monitor contains all the information related to nodes' identification and placement, a hijacking node can be easily detected and then, the mentioned node can be easily banned.

4. Simulation Framework

In order to evaluate the performance (according to various metrics) of the proposed distributed storage system for WSNs, various simulation approaches have been proposed with use of Discrete Event-based Simulators (DES) such as ns2 [19], ns3 [20], TOSSIM [21], and EmStar [22].

We have developed a DES which is based on the event-scheduling algorithm explained in [23] . In this simulator, we concentrate on events and their effects on the status of the WSN during a predetermined simulation time. Figure 4 illustrates the proposed DES structure, with the three engines described below:

1. The Simulation Engine is the heart of our DES and implements event scheduling algorithm for non-deterministic, discrete and dynamic WSNs. This engine has the following main components.
    - Snapshot, which contains: CLOCK as a base of timing system; State Variables to describe the system in different points of time; Future Event List (FEL), which contains an ordered list of future events for event scheduling algorithm execution; and Output Parameters for

evaluation and analysis. It has some functionalities in list processing for inserting events into FEL and removing events from it.
   - The event Scheduling Algorithm for event processing and updating the snapshot of the system in different points of time. The snapshot is the container of system state variables, FEL, system queues, and output counters [23].
   - Simulation Initializer, which is used to build the first snapshot of the system at the CLOCK=0.

2. The Network Engine handles all the interactions of the nodes within the WSN on the basis of the DCS data dissemination mechanisms. This engine includes the following elements.

   - GHTs for mapping an event type to a specific location. These hash functions are expedient to avoid traditional point-to-point routing approaches, which are used in today's Internet and, indeed, use data-centric approaches.
   - GPSR is routing algorithm of choice: it can route a packet toward the destination on the basis of the location of the storage node. GPSR can operate in two modes. The first mode, denoted as Greedy, is used whenever there is a node, between the current node and the destination, which is nearer to the destination than the current one. The second mode, denoted as Perimeter, is used when there is no node closer than the current one to the destination. According to the latter mode, a packet is forwarded to the neighbors of the current node based on the "right hand rule" [3] until a closer node is found. We also enhanced GPSR to route the packets more efficiently and avoid loops during the transmission of the packets by adding the list of traversed node's identification to the header of Protocol Data Units (PDUs) in Perimeter mode.
   - "Node" is the template for sensor node implementation in terms of energy and memory management.
   - "Target" is used to define target coordinations and target movements within WSN.
   - "Packet" is a container of raw information which is routed toward the storage, monitor or PDA nodes. We implement services like path tracing and loop avoidance in the structure of the packets to enhance GPSR for efficient routing and loop avoidance.
   - "Zones" are the partitions of the WSN network and are used in all considered mechanisms, i.e., R-DCS, AR-DCS, and CoMEM. Zones are associated with their boundaries and their members.

3. The two dimensional graphical engine allows to explicitly visualize events within the WSN whenever a specific algorithm is going to be used. It shows target movement, node placement, information sensing, dissemination of information toward the monitor or storage node, and the data retrieval process of PDA. It contains the following packages.

   - Board Wrapper: used to draw the boundary of the network and nodes placement.
   - Node Wrapper: used to determine wireless transmission range of the nodes and characterize transmission and reception activities.
   - Target Wrapper: indicates the target position and its movement throughout the network.
   - PDA Wrapper: chooses the position of the PDA to handle data retrieval process.
   - Packet wrapper: needs to show the ensemble of the packets within the WSN.

The DES simulator proposed in this paper does not owe to the methods we used for evaluation and analysis of energy and memory efficiency. According to its layered structure, it can easily be extended to host more applications of WSN with simple modifications. The main advantage of the mentioned simulator is related to the in dependency between network functionality and simulation graphical engines. Therefore, any modification in data dissemination protocols has no effect on the mentioned engines.

5. Analytic Results

Energy and memory efficiency are the most important evaluation factors of WSNs as they are directly related to data loss and network lifetime. In [2], [8] "Total Number of Storage Packets" ($TotalQ$),

Table 2. 802.15.4 6lowPAN network parameters.

No.	Name	Value
1	Network Size	1024 × 512 m
2	Number of Nodes	250
3	Minimum Distance	30 m
4	Number of Targets	1-16
5	Node Initial Energy	16 KJ
6	Frequency Band	2.4 GHZ
7	Bit Rate	250 Kbps
8	Transmission Energy Consumption	165 μJ
9	Reception Energy consumption	141 μJ
10	Communication Range	42 m
11	Transmission Delay	4 msec
12	Sensor Range	10 m
13	Node RAM	128 KBytes

"Total Number of Query Packets" (Q), and "Total Number of Response Packets" (Dq) are proposed as the main metrics for energy efficiency evaluation. On the other hand, in [12],[11], the "Number of Data Losses" ($Total_{dataloss}$) due to memory shortages is the main performance evaluation parameter for memory efficiency. Since the election mechanism is part of the storage process, we account for it in the traffic considered as $TotalQ$.

In order to evaluate our proposed method with the metrics outlined in the previous paragraph, we assume that sensor nodes are static – the assumption to face with the mobile nodes is under investigation. Since neighboring nodes should not be deployed too close to each other, the locations of nodes are determined randomly by considering a minimum distance. We assume that nodes are equipped with General Positioning System (GPS), so that they know their coordinates. We also assume that the quality of radio channel does not change dramatically in a short period of time, so that the transmission range is considered to be constant. Moreover, nodes are robust during storage and retrieval processes and do not crash in the presence of traffic bursts. Finally, as our research mainly focuses on the application layer, three different wireless communication technologies at physical and data-link layers: 802.11b and 802.11g, with large transmission range and relatively low node spatial density; and 802.15.4 6lowPAN, with short transmission range and relatively high node spatial density. In this paper, we concentrate on 802.15.4 6lowPAN: the extension to 802.11b and 802.11g protocols (and comparative performance analysis) will be the subject of our future work. Table 2 summarizes the input parameters for 802.15.4 6lowPAN [24]. We can categorize the metrics in Table 2 into three groups:

- Parameters depending on the application: Network Size, Number of Nodes, Minimum Distance between neighboring nodes, Number of Targets, and Sensor Range.
- Parameters based on the node specification: Node Initial Energy, which is evaluated on the basis of the initial energy of an Alkaline Battery [25]; $E_{transmit}$, computed as the product between the node voltage, the current in transmission mode, and the transmission time, i.e., 3 V × 13.8 mA × 4 ms = 165 μJ for 802.15.4 6lowPAN [24]; $E_{receive}$ computed as the product between the node voltage, the current in receiving mode, and the receiving time, i.e., 3 V × 11.8 mA × 4 ms = 141 μJ; transmission Range calculated according to Friis Law [26] with Path Loss Exponent set to 4; and Number of Memory Slots computed by dividing the available RAM for data storage in the sensor nodes by the PDU dimension.
- Parameters based on the network characteristics: Frequency Band, Bit Rate, and Transmission delay.

Table 3. Simulation parameters.

No.	Name	Value
1	Target Movement Interval	1 s
2	Target Redirection Interval	8 s
3	Sense Interval	5 s
4	Query Interval	20 s
5	Simulation Time	600 s-3600 s

In order to run our simulations, there are some parameters, related to simulation framework, that have to be evaluated as well. These parameters, shown in table 3, depend on the type of application and the simulation scenario.

In order to evaluate the performance of the storage and retrieval processes, we focus on the energy consumption of a single packet transmission and the total amount of packets which are sent in each process. On the basis of the data provided by [24], the energy consumption per packet can be calculated according to the following equation:

$$E_{packet} = E_{transmit} + E_{receive} = 165\mu J + 141\mu J = 306\mu J. \tag{7}$$

Beside $TotalQ$, Q, and Dq, to evaluate memory efficiency, we define a metric denoted as $Total_{Dataloss}$, which corresponds to the total number of packets which can not be stored in replicas due to memory shortages. According to the 802.15.4 standard, the minimum payload size is 81 Bytes: Since 40 Bytes are used for IPv6 header and 7 Bytes belongs to UDP header, 33 Bytes lay for data. The use of JSON data structure consumes 7 more Bytes, so that the space left for key/value pair (the "data") is 26 Bytes. Therefore, the amount of data loss can be computed as follows:

$$DataLoss = Total_{dataloss} \times 26\text{Bytes}. \tag{8}$$

5.1. Short-term Simulations

In short term simulations, we evaluate each data storage algorithm by inserting different number of targets in the simulated WSN. The number of targets is between 1 and 120 in order to force different levels of storage and retrieval traffic in a short period of time (600 s). For long-term simulations, we set the number of targets in the range between 1 and 50—the reason for this change is related to the amount of packets generated by the simulator and will be discussed later in Section 5.2.

5.1.1. TotalQ

Obviously, $TotalQ$ is minimized when using LCDDR, as data is stored locally without any transmission. Therefore, the traffic due to storage is only limited to the election mechanism. The basic DCS ranks second in terms of storage traffic, as it only relies on storage nodes and no fault tolerance mechanism is implemented: therefore, it does not contain any election mechanism for storage and this also decreases the traffic due to storage. R-DCS ranks third. With this algorithm, monitors are also used in the storage process: this creates more traffic and makes R-DCS vulnerable to monitor failure. Finally, CoMEM is the method with monitor and storage nodes collaborating in the storage process and uses two election mechanisms for monitor and storage node election. Therefore, the traffic amount for storage is relatively higher with respect to other methods. In Figure 5, the performance results, in terms of $TotalQ$, as a function of the number of the targets, are shown. As the results in Figure 5 show, there is no election traffic with LCDDR, so that the election mechanism is not relevant in short-term simulations.

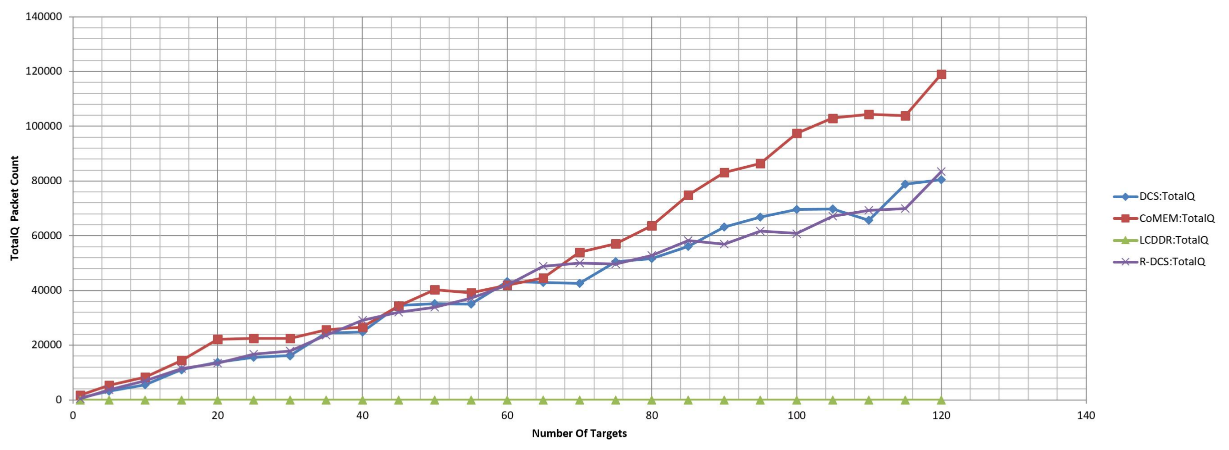

Figure 5. TotalQ in short-term simulation

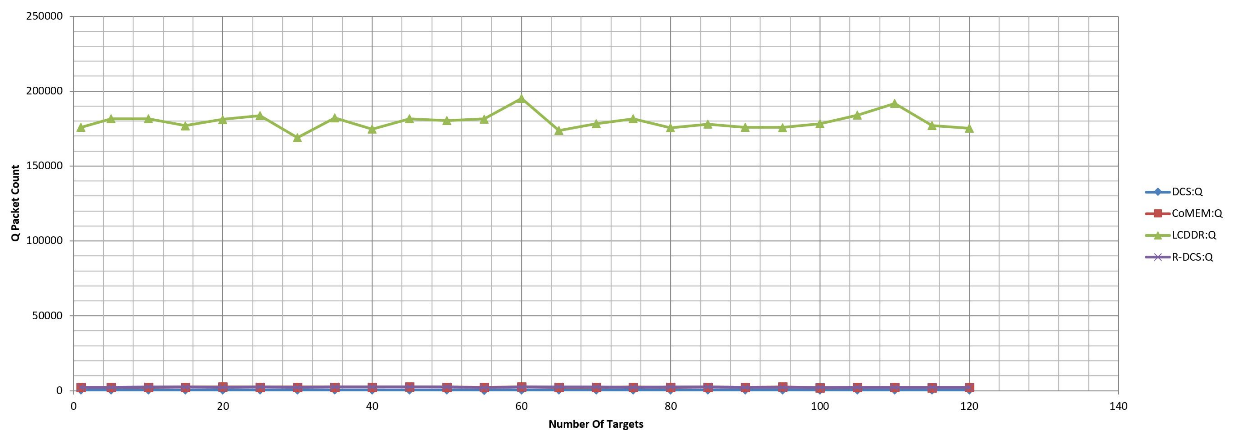

Figure 6. Q in short-term simulation

5.1.2. Q and Dq

The metrics Q and Dq are expedient in understanding the efficiency of the retrieval process. Figure 6 and Figure 7 show the performance results, respectively, in terms of Q and Dq as functions of the number of targets. The most efficient retrieval process belongs to the basic DCS, because only storage nodes are involved in queries and responses. However, this mechanism suffers from data loss due to node failure. The second position belongs to R-DCS and CoMEM due to the role of the monitor as a gateway in retrieval process. R-DCS also suffers from the single point of failure risk related to the energy depletion of the monitor. Finally, the LCDDR mechanism faces the highest amount of traffic due to its inefficient retrieval process: this brings traffic bursts in the retrieval process and decreases the network lifetime.

5.1.3. Total Number of Packets

As shown in Figure 8, CoMEM traffic is relatively increased, with respect to DCS and RDCS, because of the election mechanism. On the other hand, LCDDR mechanism incurs a high amount of traffic and, consequently, a relevant energy consumption (approximately 75 J over 600 s). It can thus be concluded that CoMEM brings fault tolerance and load-balancing to data storage and retrieval processes, keeping the traffic close to that of the basic distributed storage methods.

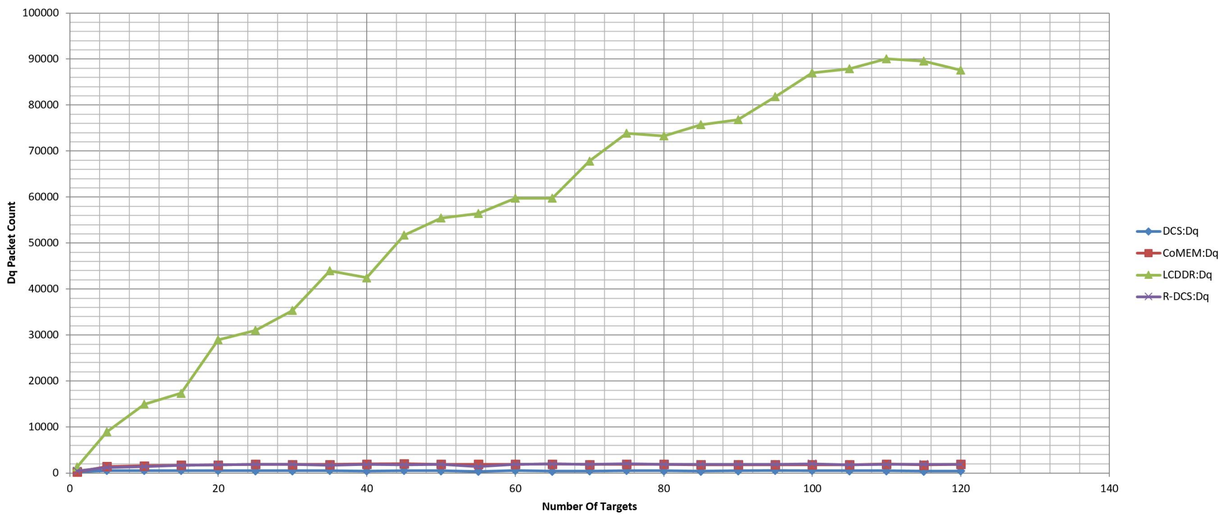

Figure 7. Dq in short-term simulation

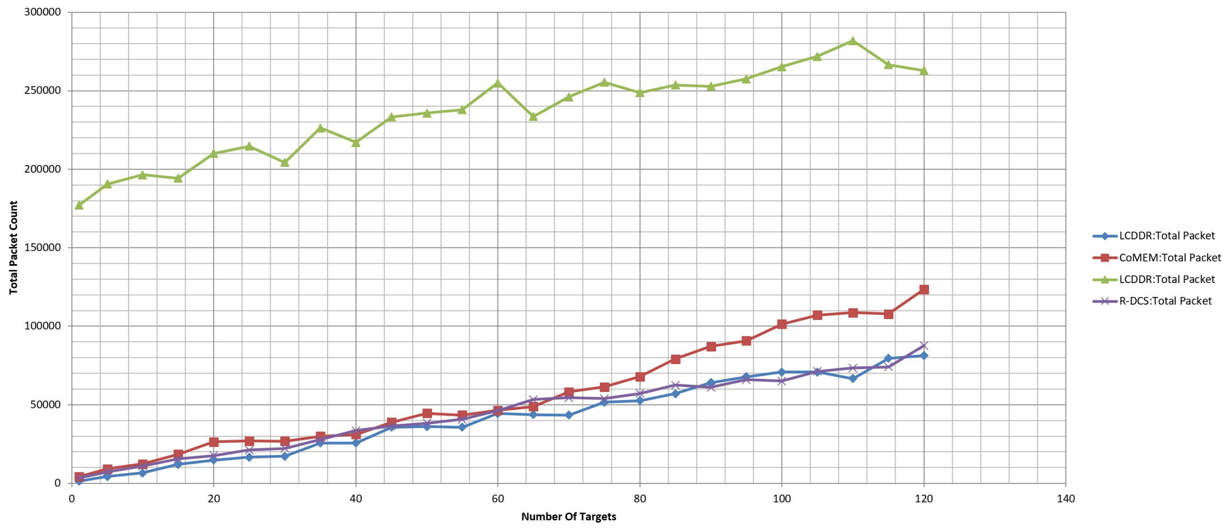

Figure 8. Total Number of Packets in short-term simulation

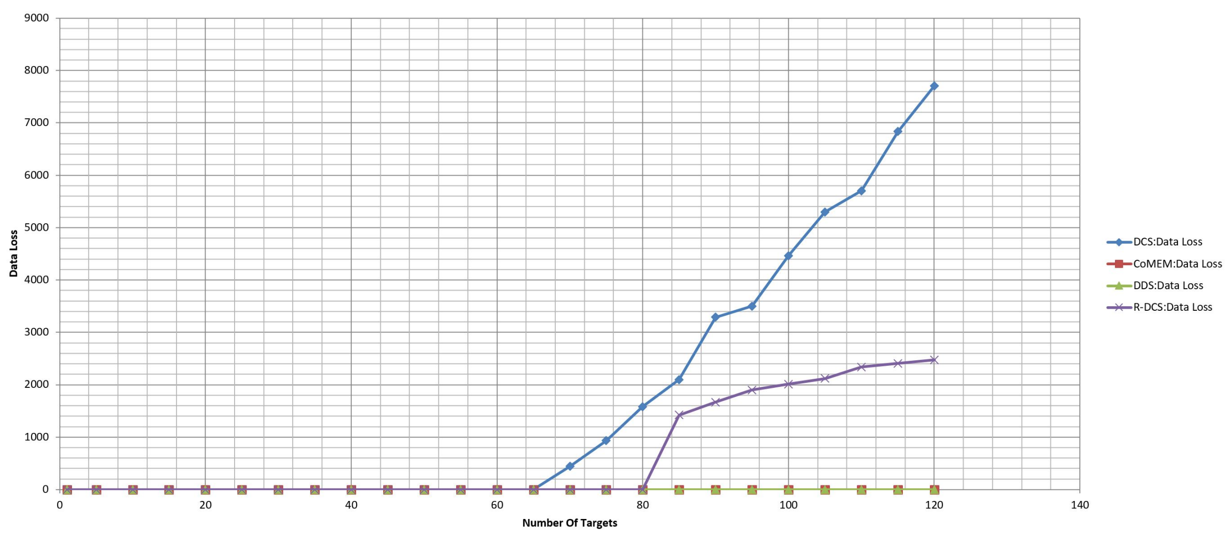

Figure 9. Data loss in short-term simulation

5.1.4. $Total_{dataloss}$

In Figure 9, $Total_{dataloss}$ is shown as a function of the number of targets. As expected, the amount of data loss in basic data dissemination methods is high, whereas LCDDR and CoMEM suffer no data loss in short-term simulations.

5.2. *Long-term Simulations*

In order to enhance our results, we developed a low-level (taking into account hardware constraints) DES simulator based on the framework presented in Section 4 over 32 bit MinGW engine [27]. In this simulator, we use C++ compiler and omit 2 dimension animation to enhance the simulation speed. With this simulator, we can expand the simulation time from 600 s to 3600 s to collect more accurate performance results. However, the main problem with 32 bit MinGW engine is related to the amount of RAM which can be accessed during the simulation. This value corresponds to 4 GB with the used computer. Therefore, applications which generate a large amount of packets, such as LCDDR, can not be simulated with our simulator. Moreover, we can not consider a large number of targets as in short-term simulations. More precisely, the number of targets is set in a range between 1 and 50. By using a large memory server and a 64 bit compiler engine, these limitations disappear. This extension is the topic of our future research.

5.2.1. TotalQ

In long-term simulations, since the neighbors of the storage nodes begin to deplete their energies, the paths to reach storage nodes become longer. Therefore, the traffic associated with the storage process when using the basic methods reaches the value of the traffic load of CoMEM. In Figure 10, $TotalQ$ is shown as a function of the number of targets, confirming this observation. As a result, the traffic due to the storage process in long-term simulations is relatively the same for CoMEM, DCS, and R-DCS.

5.2.2. Q

The traffic due to the query process depends completely on the storage algorithm in data-centric storage methods. More complicated storage algorithms incure a higher query traffics. As shown in Figure 11 and Figure 12, the lower traffic is guaranteed by DCS, which has the simplest data dissemination mechanism. CoMEM and R-DCS have a similar performance.

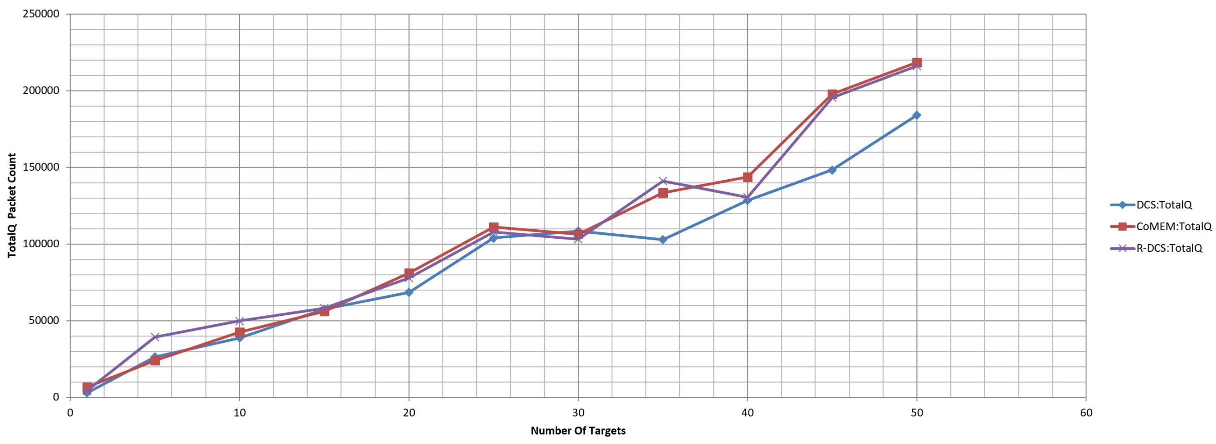

Figure 10. TotalQ in long-term simulation

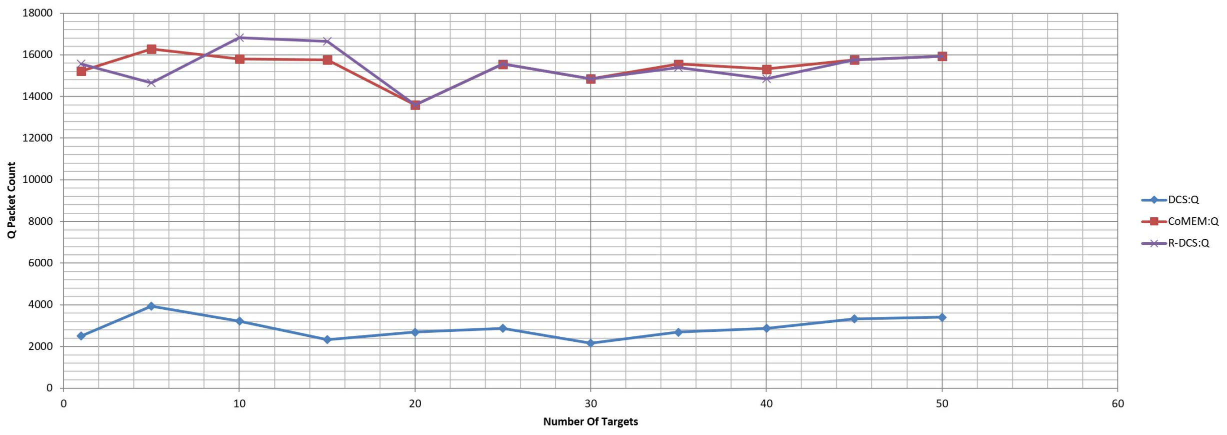

Figure 11. Q in short-term simulation

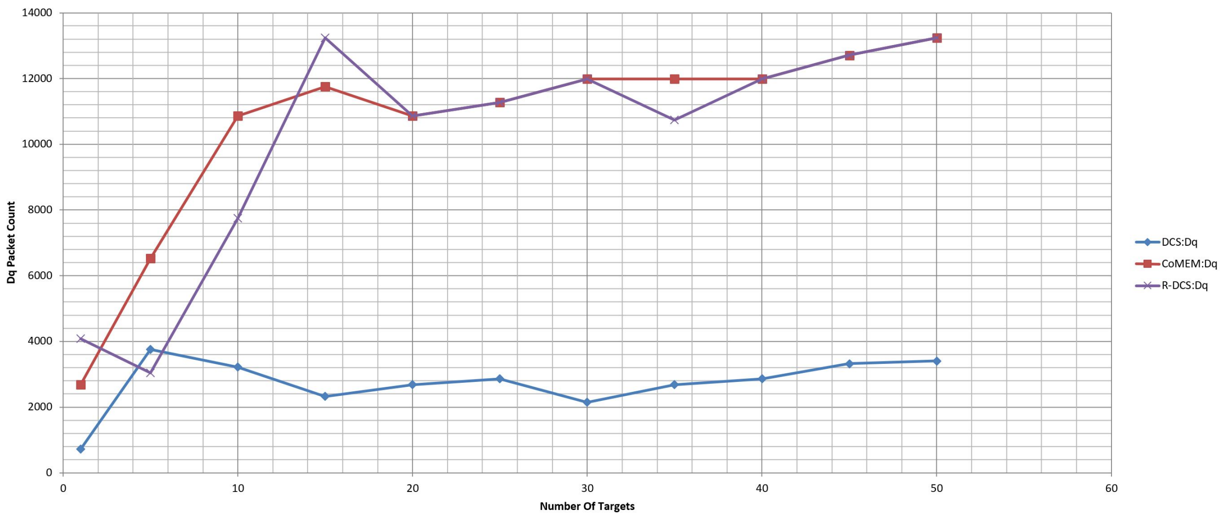

Figure 12. Dq in long-term simulation

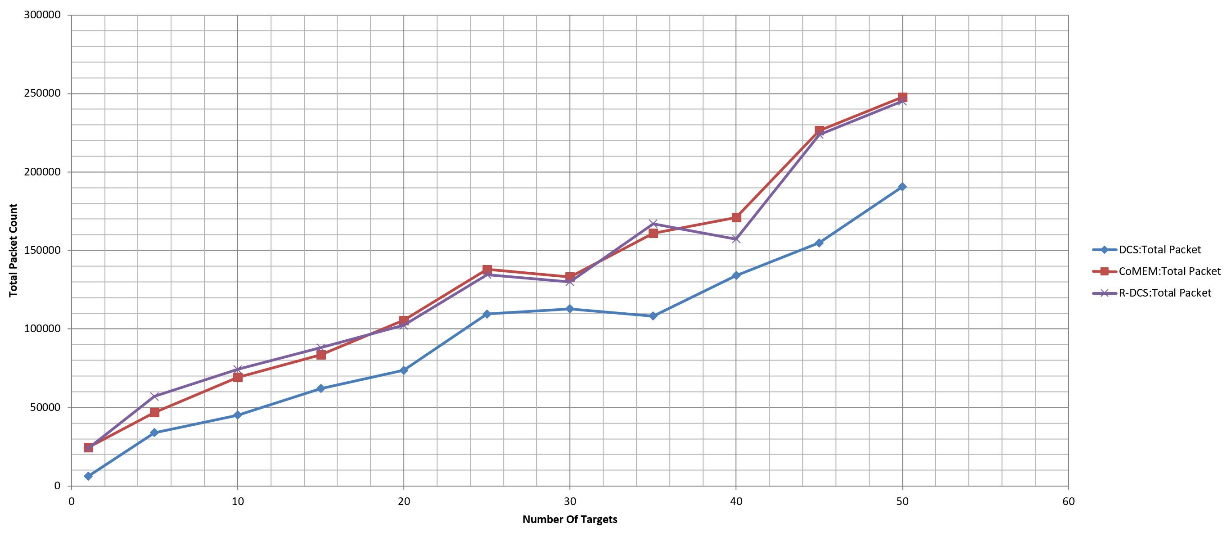

Figure 13. Total Number of Packets in long-term simulation

5.2.3. Total Number of Packets

In Figure 13, the total number of packets is shown as a function of the number of targets. Since the traffic due to the storage process is approximately the same in the three considered methods, the total number of packets is directly affected by the query process traffic. For CoMEM, the total energy consumption is approximately 75 J, which is equal to the amount of energy consumption in LCDDR in short-term simulations . Therefore, it can be concluded that CoMEM is six times more energy efficient than LCDDR.

5.2.4. $Total_{Dataloss}$

In Figure 14, the amount of data loss is shown as a function of the number of targets. In order to reduce to zero the data loss incured by CoMEM, we change the value of C1 and C2 parameters presented in Section 3. Since the initial energy associated with the nodes is quite high (16 KJ) with respect to the energy consumption associated with the transmission and reception of each packet (300 μJ), we set C1 to 0.1 and C2 to 0.9 in order to give a higher priority to memory efficiency.

According to the results in Figure 14, CoMEM is incurs no data loss in the long term simulation. It also generates load balancing to the traffic due to storage process over multiple replicas. Moreover, the possibility of setting $C1$ and $C2$ makes it flexible and able to adapt to various situations.

6. Conclusions

In both DCS and DDS singular mechanisms, nodes are reliable and the network topology does not change over time, so that node failure has not been considered. However in the presence of node failure, these assumptions no longer hold. In this paper, a novel scheme denoted as CoMEM is proposed which integrates the DCS features into DDS. CoMEM avoids local broadcasts for storage election and brings energy and memory efficiency to the data loss prevention mechanism for data dissemination algorithms in WSNs. CoMEM uses the features of DCS in storage and retrieval processes: therefore, it does not generate a high amount of traffics in the network. However, it collaboratively prevents data loss or traffic failure due to nodes failure.

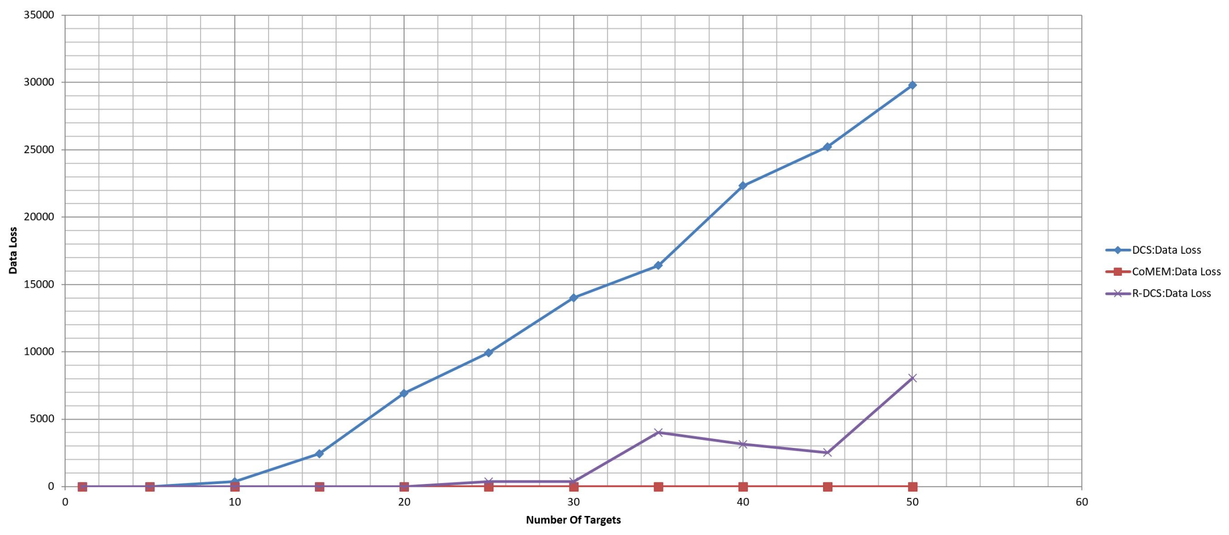

Figure 14. Data loss in long-term simulation

7. References

1. Yick, J.; Mukherjee, B.; Ghosal, D. Wireless sensor network survey. *Computer networks* **2008**, *52*, 2292–2330.
2. Ratnasamy, S.; Karp, B.; Yin, L.; Yu, F.; Estrin, D.; Govindan, R.; Shenker, S. GHT: a geographic hash table for data-centric storage. Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications. ACM, 2002, pp. 78–87.
3. Karp, B.; Kung, H.T. GPSR: Greedy perimeter stateless routing for wireless networks. Proceedings of the 6th annual international conference on Mobile computing and networking. ACM, 2000, pp. 243–254.
4. Aly, M.; Pruhs, K.; Chrysanthis, P.K. KDDCS: a load-balanced in-network data-centric storage scheme for sensor networks. Proceedings of the 15th ACM international conference on Information and knowledge management. ACM, 2006, pp. 317–326.
5. Lai, Y.; Wang, Y.; Chen, H. Energy-efficient robust data-centric storage in wireless sensor networks. Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on. IEEE, 2007, pp. 2735–2738.
6. Cuevas, A.; Urueña, M.; Veciana, G.; Cuevas, R.; Crespi, N. Dynamic data-centric storage for long-term storage in wireless sensor and actor networks. *Wireless networks* **2014**, *20*, 141–153.
7. Ahmed, K.; Gregory, M.A. Distributed efficient similarity search mechanism in wireless sensor networks. *Sensors* **2015**, *15*, 5474–5503.
8. Chuang, A.G.J.G.J. Resilient Data-Centric Storage in Wireless Sensor Networks. *IEEE Distributed Systems Online* **2003**, *4*.
9. Hejazi, P. An adaptive hybrid schema for data-centric storage in wireless sensor networks. *International Journal of Distributed Sensor Networks* **2016**, *12*.
10. Dudkowski, D.; Marron, P.J.; Rothermel, K. An efficient resilience mechanism for data centric storage in mobile ad hoc networks. Mobile Data Management, 2006. MDM 2006. 7th International Conference on. IEEE, 2006, pp. 7–7.
11. Gonizzi, P.; Ferrari, G.; Gay, V.; Leguay, J. Data dissemination scheme for distributed storage for IoT observation systems at large scale. *Information Fusion* **2015**, *22*, 16–25.
12. Amiri, S.M.S.; Malazi, H.T.; Ahmadi, M. Memory Efficient Routing Using Bloom Filters in Large Scale Sensor Networks. *Wireless Personal Communications* **2016**, *86*, 1221–1240.

13. Xu, H.; Li, Y.; Collinge, W.O.; Schaefer, L.A.; Bilec, M.M.; Jones, A.K.; Landis, A.E. Improving efficiency of wireless sensor networks through lightweight in-memory compression. Green Computing Conference and Sustainable Computing Conference (IGSC), 2015 Sixth International. IEEE, 2015, pp. 1–8.
14. Chouikhi, S.; El Korbi, I.; Ghamri-Doudane, Y.; Saidane, L.A. A survey on fault tolerance in small and large scale wireless sensor networks. *Computer Communications* **2015**, *69*, 22–37.
15. Banerjee, J.; Mitra, S.K.; Ghosh, P.; Naskar, M.K. Memory based message efficient clustering (MMEC) for enhancement of lifetime in wireless sensor networks using a node deployment protocol. Proceedings of the 2011 International Conference on Communication, Computing and Security. ACM, 2011, pp. 71–76.
16. Zhou, H.; Guan, X.; Wu, C. Reliable transport with memory consideration in wireless sensor networks. 2008 IEEE International Conference on Communications. IEEE, 2008, pp. 2819–2824.
17. Piotrowski, K.; Langendoerfer, P.; Peter, S. tinyDSM: A highly reliable cooperative data storage for Wireless Sensor Networks. Collaborative Technologies and Systems, 2009. CTS'09. International Symposium on. IEEE, 2009, pp. 225–232.
18. Chessa, S.; Maestrini, P. Fault recovery mechanism in single-hop sensor networks. *Computer communications* **2005**, *28*, 1877–1886.
19. NS2. http://www.isi.edu/nsnam/ns/. Accessed: 2017-01-08.
20. NS3. http://www.isi.edu/nsnam/ns/],ns3[https://www.nsnam.org/. Accessed: 2017-01-08.
21. TOSSIM. http://tinyos.stanford.edu/tinyos-wiki/index.php/TOSSIM. Accessed: 2017-01-08.
22. EmStar. https://www.usenix.org/legacy/event/usenix04/tech/general/full_papers/girod/girod_html/eu.html. Accessed: 2017-01-08.
23. Jerry, B. *Discrete-event system simulation*; Pearson Education India, 1998.
24. Atmel-8351. http://www.atmel.com/images/Atmel-8351-MCU_Wireless-AT86RF233_Datasheet.pdf. Accessed: 2017-01-08.
25. Alkaline Battery Specification. https://en.wikipedia.org/wiki/AA_battery. Accessed: 2017-01-08.
26. Friis Law. https://en.wikipedia.org/wiki/Friis_transmission_equation. Accessed: 2017-01-08.
27. Minimalist GNU for Windows. http://http://www.mingw.org. Accessed: 2017-07-22.