electronics

MDPI

Article

Novel Extensions to Enhance Scalability and Reliability of the IEEE 802.15.4-DSME Protocol

Filippo Battaglia [1,†], Mario Collotta [2,†], Luca Leonardi [1,†], Lucia Lo Bello [1,†] and Gaetano Patti [1,*,†]

[1] Department of Electrical, Electronics and Computer Engineering, University of Catania, Viale Andrea Doria 6, 95125 Catania, Italy; fbattaglia@unict.it (F.B.); luca.leonardi@unict.it (L.L.); lobello@unict.it (L.L.B.)

[2] Faculty of Engineering and Architecture, Kore University of Enna, 94100 Enna, Italy; mario.collotta@unikore.it

* Correspondence: gaetano.patti@unict.it; Tel.: +39-095-738-2386

† All the authors contributed equally to this work.

Received: 12 December 2019; Accepted: 6 January 2020; Published: 9 January 2020

Abstract: The Deterministic and Synchronous Multichannel Extension (DSME) of the IEEE 802.15.4 standard was designed to fulfill the requirements of commercial and industrial applications. DSME overcomes the IEEE 802.15.4 limitation on the maximum number of Guaranteed Time Slots (GTS) in a superframe and it also exploits channel diversity to increase the communication reliability. However, DSME suffers from scalability problems, as its multi-superframe structure does not efficiently handle GTS in networks with a high number of nodes and periodic flows. This paper proposes the enhanceD DSME (D-DSME), which consists of two extensions that improve the DSME scalability and reliability exploiting a GTS within the multi-superframe to accommodate multiple flows or multiple retransmissions of the same flow. The paper describes the proposed extensions and the performance results of both OMNeT simulations and experiments with real devices implementing the D-DSME.

Keywords: IEEE 802.15.4; DSME; Real-time networks; Industrial Wireless Sensor Networks

1. Introduction

Presently, a wide range of industrial and automation applications adopt Industrial Wireless Sensor Networks (IWSNs) to benefit from the deployment flexibility and the support for a large number of mobile nodes that these networks offer. In this context, the IEEE 802.15.4 standard [1] is one of the most widely used technologies, as it supports different kinds of traffic through different medium access strategies, such as TDMA, CSMA/CA and prioritized channel access (PCA) [2]. In 2012, the IEEE 802.15.4e [3] amendment introduced three new Medium Access Control (MAC) protocols, namely the Deterministic and Synchronous Multichannel Extension (DSME), the Time Slotted Channel Hopping (TSCH) [4] and the Low Latency Deterministic Network (LLDN) [5]. Later, DSME and TSCH were included in the IEEE 802.15.4-2015 standard.

DSME is a MAC protocol designed for critical application domains with stringent requirements, such as, deterministic delay, high reliability, and adaptability to changes in the network traffic and operating conditions. DSME uses frequency hopping to mitigate the effects of external interference [6].

As discussed in [7,8], DSME is particularly suitable for several industrial, commercial and healthcare applications, such as factory automation, home automation, smart metering, smart buildings, and patient monitoring. DSME can be successfully used in solar tower power plant industries [9], in WSN applications such as outdoor monitoring (as those reported in [10,11]) and in other

applications [12] in which the number of network nodes or the generated data flows change over time [7] in response to an external event. For instance, the work in [13] discusses the use of DSME in surveillance systems, such as temperature reading or low-resolution images and video, in which devices send a small amount of data. In particular, in the addressed use cases, the bitrate of a video stream increases upon movement detection and the sample time of a pollution monitoring system decreases when the guard levels are exceeded.

DSME uses a tree topology made up of a root node (named the *PAN-Coordinator*), one or multiple *coordinators*, and one or multiple *end-nodes* [1]. The end-nodes receive (transmit) data only from (to) their own coordinators. The end nodes use beacons to synchronize with the network time.

The network time is divided into multiple *beacon intervals*, each one containing $m = 2^{BO-MO}$ multi-superframes. Each multi-superframe is divided, in turn, into $n = 2^{MO-SO}$ superframes. The parameters BO, MO and SO (named *Beacon order*, *Multi-superframe order* and *Superframe order*, respectively) are configurable. Each superframe, as it is shown in Figure 1, includes the beacon slot and two sections for data transmission, i.e., the Contention Access Period (CAP) and the Contention-Free Period (CFP).

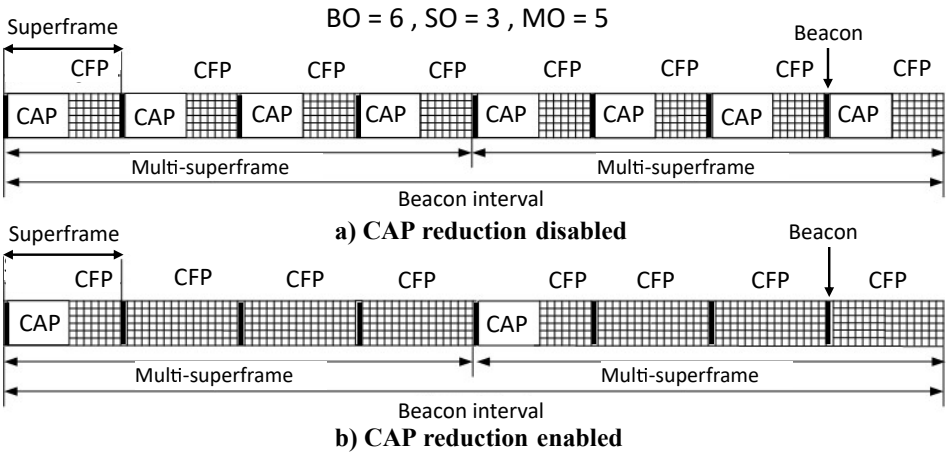

Figure 1. DSME multi-superframe structure.

The CAP is used to transmit the aperiodic traffic. During the CAP, no scheduling of source-destination node pairs is used, therefore each node can start the transmission according to the CSMA/CA access mechanism. Conversely, the CFP is divided into multiple GTSs that are used to transmit periodic traffic in a TDMA-like strategy. The duration of the CAP and of the CFP slots is customizable.

DSME provides diverse additional options, such as the CAP reduction and the Group Acknowledgement (ACK). The former enables CAP only in the first superframe of a multi-superframe, thus providing the remaining superframes in the multi-superframe with a longer CFP, i.e., 15 GTSs instead of 7. The latter option, instead, allows aggregating the ACK of multiple frames into a single ACK frame, thus improving the energy efficiency. The Group ACK also allows specifying a GTS for the retransmissions of the frames not correctly received and reduces the delay, as it provides a retransmission opportunity within the same multi-superframe in which the original frame was transmitted.

Motivation. In DSME, each GTS within the multi-superframe can be assigned to at most one flow. If the flow period is longer than the multi-superframe period, bandwidth waste occurs, as some of the GTSs reserved to the flow will not be used. Moreover, the rigid structure of the multi-superframe impairs scalability [4], because in DSME the number of superframes can be set only to powers of 2 (i.e., 1, 2, 4, 16, and so on). As a result, the multi-superframe length exponentially grows, and, with the CAP reduction enabled, this results in a low CAP frequency (i.e., one CAP every multi-superframe) and, therefore, in high delays in the transmission of the aperiodic messages. A preliminary approach to

cope with the DSME scalability problem is addressed in the work [14] which, however, provide neither a detailed design of the approach nor an extensive performance evaluation. In addition, the work in [14] does not address the DSME reliability.

Contributions. This work proposes the enhanceD-DSME (D-DSME) extensions, i.e., two novel extensions that improve the DSME scalability and reliability. The first one, named *Shareable D-DSME*, allows GTSs to be shared among multiple periodic flows when the flow period is longer than the multi-superframe period. This improves scalability, as more flows can be scheduled without increasing the multi-superframe size and, therefore, without increasing the average delays for the aperiodic messages transmitted in the CAP, as the CAP frequency obtained using the D-DSME is higher than or equal to the one in the standard DSME protocol.

The second extension, named *Mirror D-DSME*, allows using the same GTSs for both message transmission and retransmissions. This feature improves reliability, as a message is retransmitted multiple times.

Henceforward, we will use the D-DSME acronym to refer to both the extensions, i.e., the Shareable and the Mirror DSME.

Organization. The paper is organized as follows. Section 2 deals with related work. Section 3 describes the Shareable D-DSME extension, while Section 4 presents the Mirror D-DSME extension. Section 5 provides a comparative performance assessment of DSME and D-DSME. Section 6 presents a proof-of-concept implementation of D-DSME that demonstrates the feasibility of the proposed extensions. Finally, Section 7 gives conclusions and hints for future work.

2. Related Work

Several works addressed the features and performance of the IEEE 802.15.4 standard. For instance, the works in [15,16] address the real-time capabilities of IEEE 802.15.4, while in [5,17] multichannel approaches to improve the protocol scalability were proposed.

Recently, thanks to its ability to cope with timing and reliability requirements, the DSME protocol is gaining ground over the plain IEEE 802.15.4 for general industrial and commercial applications. Consequently, several researchers showed high interest in assessing the protocol and in devising solutions to further improve its performance. As far as interference mitigation is concerned, a DSME device in channel adaptation mode switches to another available channel if the received signal quality is less than a threshold value [18,19]. The work in [19] considered a single interfering IEEE 802.11b (WiFi) source and evaluated the error frame rate introduced in a DSME network, for different traffic loads and power levels of the WiFi transmission. As the simulation results show, the frame error rate cannot exceed 25% in DSME, since a WiFi channel can overlap with at most four (out of 16) of the channels used in DSME. Hence, DSME can exploit the channels that are not affected by WiFi transmissions to communicate without interference. Other approaches in the literature, such as [20], propose a new superframe structure in order to mitigate the problems of WiFi and ZigBee interference. As far as throughput and energy consumption are concerned, the assessment in [18] shows that the DSME protocol with the CAP reduction enabled performs better than the plain IEEE 802.15.4 protocol. The works in [21,22] introduce some enhancements for DSME to reduce both the energy consumption of the end nodes during the CAP and the network discovery time.

In [23] the authors propose a novel DSME access scheme that divides the network nodes in multiple subsets, each one associated with a CAP slot in the beacon interval. During each CAP slot, only the nodes in the related subset perform channel contention. This way, the probability of collision between transmitting nodes is reduced and a lower number or retransmissions are required. In addition, an improved slot allocation scheme for CFP section is adopted. When a GTS transmission fails, the next CAP slot within the multi-superframe is used for retransmission. The proposed access scheme can reduce delay and power consumption. However, it is intended only for a star topology (not for a mesh topology). The work in [24] addresses the integration of RPL (IPv6 Routing Protocol for Low Power and Lossy Networks) and proposes Symphony for DSME, a dynamic GTS scheduling

algorithm that integrates RPL over DSME to provide a QoS efficient schedule for GTS placement. However, this approach implies that the network schedule is repeated periodically. Thus, every time a cyclic-scheduling allocation problem needs to be solved. In addition, the devices must implement the protocol stack proposed in [24].

A new DSME implementation, called openDSME, is presented in [25]. The authors also propose a method for traffic-aware and decentralized slot scheduling in order to realize scalable IWSNs. It is a versatile solution, as it does not require routing information and therefore it can be used with any routing protocol. The works in [26,27] propose a scheduling algorithm, implemented using RPL integration and OpenDSME, which changes dynamically the multi-superframe structure as a function of the current number of network nodes, in order to reduce transmission delays.

The work in [28] proposes a DSME-based distributed scheduling algorithm for mobility support that adaptively assigns GTSs based on the network traffic of each node, thus improving the network reliability and timeliness.

Differently from D-DSME, the solutions proposed in [13,23–28] do not address any sharing mechanism to improve the GTSs use, i.e., the ratio between the number of GTSs actually used and the number of the assigned ones, when the flow period is larger than the multi-superframe one. Moreover, they do not support unconfirmed retransmission of multiple replicas of the same message in order to improve reliability. Recently, the growing interest in DSME has also fostered the development and release of open-source implementations. Among them there are [9,25,29].

3. Shareable D-DSME Extension

In the DSME multi-superframe, each node is assigned one or multiple GTSs for the transmission of messages belonging to one flow. In Figure 2 the multi-superframe contains two superframes ($T_{MSF} = 2T_{SF}$). There are 15 periodic real-time flows, 8 of them $1 \ldots 8\}$ with period $P_{f[1...8]} = 4T_{SF} > T_{MSF}$ and 7 of them $\{9 \ldots 15\}$ with period $P_{f[9...15]} = 2T_{SF} = T_{MSF}$. In the plain DSME schedule, 15 GTSs are assigned for these flows within one multi-superframe. The slots assigned to the flows $\{1 \ldots 8\}$ are not always used, as $P_{f[1...8]}$ is larger than the multi-superframe duration T_{MSF}.

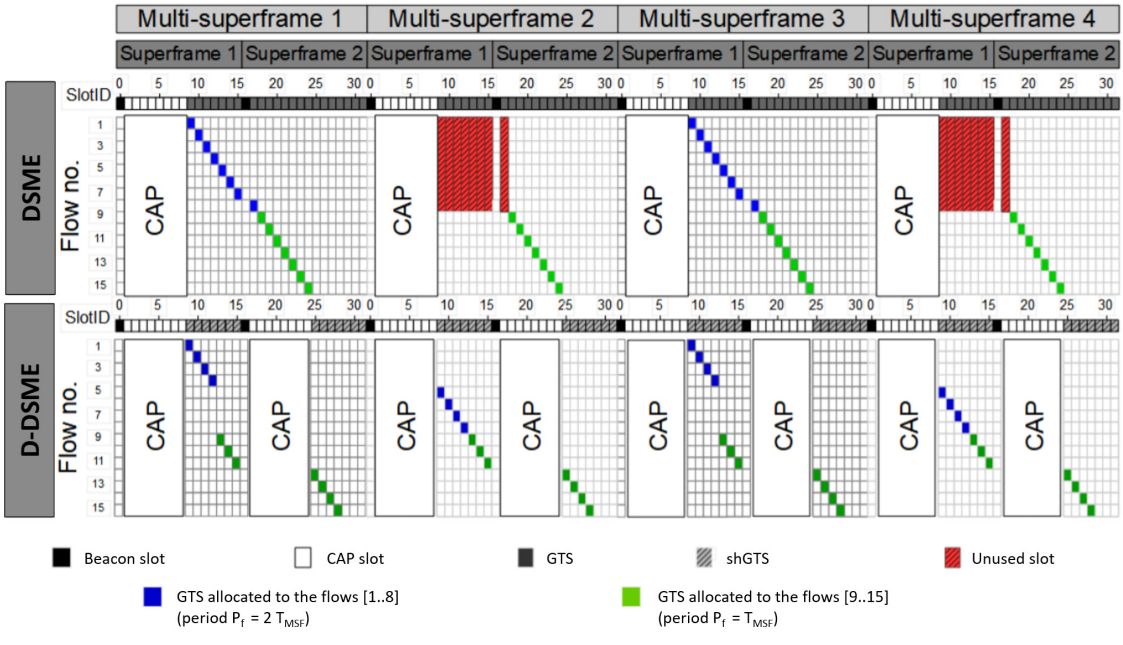

Figure 2. Scheduling example using DSME and D-DSME.

The *Shareable D-DSME* extension introduces the *Shareable GTS (shGTS)*, which can be shared among multiple periodic flows, i.e., the ones with a period $P_f > T_{MSF}$, transmitted by the same or

different nodes. Thanks to the *shGTS*, *Shareable D-DSME* can schedule the same flows as DSME using a lower number of GTSs. This features results in more free CFP slots that can be used either to enable the CAP in each superframe (as shown in the example in Figure 2) or to accommodate new periodic flows.

In the D-DSME schedule shown in Figure 2, the *shGTSs* with index $\{9 \dots 12\}$ are assigned to the flows $\{1 \dots 4\}$ in the MSF 1 and to the flows $\{5 \dots 8\}$ in the MSF 2. This way, the flows 5-8 can be transmitted in the same GTSs assigned to flows 1-4, but with an offset (i.e., a shift) of T_{MSF}. Instead, the GTSs $\{13 \dots 15\}$ and $\{25 \dots 28\}$ are assigned in every MSF to the flows $\{9 \dots 11\}$ and $\{12 \dots 15\}$, respectively. No sharing is possible for these flows, as $P_{f[9 \dots 15]} = T_{MSF}$.

The D-DSME does not restrict the applicability of DSME to centralized architectures, as it also support mesh topologies. The two following subsections describe the DSME and the D-DSME allocation procedures, respectively.

3.1. Plain DSME Allocation Procedure

The GTSs in the CFP are allocated using a distributed procedure [1], suitable for operating in a mesh network. In order to support the operation, each node maintains two items in memory:

- the *macDSMEACT table*, which contains all the slots in the multi-superframe in which the node is source or destination;
- the *macDSMESAB mask*, which contains all the slots in the multi-superframe that the node knows to be busy.

When a node X needs to allocate a slot to transmit to a destination node Y, it sends to Y a *DSME_GTS_REQUEST* message (hereinafter named M_{req}) containing:

- the item macDSMESAB(X), i.e., the current *macDSMESAB mask* of node X;
- the identifier of the slot that node X requests to allocate.

The node Y performs the OR operation between its own current macDSMESAB(Y) mask and the macDSMESAB(X) field in the received M_{req}. If the bit corresponding to the required slot in the OR-ed result is unset, there is no conflict (at least, in the knowledge of X and Y) and Y can allocate the slot to X. In such a case, the destination node Y updates its local *macDSMESAB(Y)* mask and sends in broadcast a *DSME_GTS_RESPONSE* message (hereinafter named M_{resp}) to all the other nodes, indicating the address of the source node X and the id of the newly allocated slot.

When the node X receives M_{resp} from Y, updates its local *macDSMESAB(X) mask* and *macDSMEACT(X) table*. Next, the node X sends a *DSME_GTS_NOTIFY* message (hereinafter named M_{not}) in broadcast to all the other nodes, indicating the address of Y and the id of the newly allocated slot.

The M_{resp} and M_{not} messages are sent broadcast. This way, all the neighbours of the nodes X and Y are informed that a new slot was allocated, so they can mantain a consistent view of the busy slots. When a node Z in the neighbourhood of X and Y receives M_{resp} or M_{not}, it verifies that there is no conflict on the assigned slot using its local *macDSMESAB(Z) mask*. If no conflict is detected, the mask is updated. Otherwise, the node Z sends a message to the other nodes and starts a new procedure to handle the conflict.

3.2. EnhanceD DSME Allocation Procedure

In a mesh network running D-DSME, each node sends the allocation request to its nearest coordinator, which re-routes the message (through multi-hop transmissions) towards the *PAN-Coordinator*. Later, the allocation response generated by the *PAN-Coordinator* is sent to one *coordinator*, which re-routes the message (also, in a multi-hop way) towards the destination node. Only scheduling is centralized, as GTS allocation is up to the *PAN-Coordinator*.

Table 1 summarizes the notation used in this subsection.

Table 1. Summary of notation.

Symbol	Definition
T_{MSF}	Multi-superframe duration
T_{SF}	Superframe duration
n_{SF}	Number of superframes a multi-superframe
P_i	Period of the i-th flow
$shGTSInterval$	Interval between two consecutive transmissions of a message belonging to a flow f, expressed in number of multi-superframes.
$shGTSOffset$	Offset in terms of multi-superframes assigned to a flow
$\mathcal{F}_s$	Set of flows assigned to the *shGTS* s
$n_{\mathcal{F}_s}$	Number of flows that share the *shGTS* s
$MSFId$	Multi-superframe id
$SFId$	Index of a superframe in a beacon interval

The D-DSME allocation procedure is performed in four steps, described as follows:

Step 1. When a node needs to allocate a GTS for a flow f, it sends a request to the *PAN-Coordinator* indicating the flow period. For each *shGTS* the *PAN-Coordinator* checks if, in the case the new flow is assigned the s-th *shGTS*, the slot use ($U(s)$) is lower than or equal to 1, i.e.,

$$U(s) = T_{MSF} \sum_{i \in \mathcal{F}_s} \frac{1}{P_i} \leq 1 \tag{1}$$

where, $\mathcal{F}_s$ are the flows assigned to the *shGTS* s (including the new flow f), P_i is the period of the i-th flow and T_{MSF} is the multi-superframe duration. The slot use indicates the ratio of used GTS during the time. For instance, if a flow has period 10s and the multi-superframe duration is 2s, only one GTS out of five will be used for transmitting that flow.

Step 2. If condition (1) is met, the *PAN-Coordinator* tests if an *offset* (in terms of the number of multi-superframes), for each flow that shares the *shGTS* s, can be found so as to avoid mutual interference. Hence, it checks if the greatest common divisor (GCD) of the flow periods that share the *shGTS* s is greater than or equal to the number of flows that share the *shGTS* s ($n_{\mathcal{F}_s}$) multiplied by the multi-superframe duration (T_{MSF}), i.e.,

$$\mathbf{GCD}(P_i) \geq n_{\mathcal{F}_s} \times T_{MSF}, \forall i \in \mathcal{F}_s \tag{2}$$

If condition 2 holds, then the *shGTS* s can be shared among all the flows belonging to $\mathcal{F}_s$ and consecutive offsets starting from 0 can be assigned to these flows. The *PAN-Coordinator* assigns to the flow f the next consecutive offset *shGTSOffset*, calculated as $shGTSOffset = n_{\mathcal{F}_s} - 1$. If no *shGTS* meets both the conditions (1) and (2), either the flow will be assigned an unassigned GTS, if any is found, or the allocation request will be rejected.

Step 3. Once a *shGTS* is found for the new flow f and the offset *shGTSOffset* is assigned, the *PAN-Coordinator* determines the shared GTS Interval *shGTSInterval*, i.e., the interval between two consecutive transmissions of a message belonging to the flow f, expressed in number of multi-superframes. For example, in Figure 2, the offset *shGTSOffset* of the flow 5 is equal to 1 (i.e., one multi-superframe) and its *shGTSInterval* is equal to 2 (i.e., the flow 5 can be transmitted in its GTS every two multi-superframes). The *shGTSInterval* is calculated as in the following formula:

$$shGTSInterval = \left\lfloor \frac{P_f}{T_{MSF}} \right\rfloor \tag{3}$$

Step 4. The *PAN-Coordinator* sends the GTS allocation response to the node indicating the *FlowIdentifier* f, the *shGTSId* (i.e., slot id and superframe id of the assigned *shGTS* within the multi-superframe), the assigned *shGTSOffset* and the *shGTSInterval*. Once the response is received, if the allocation was successful, the node requesting the GTS updates its own *macDSMESAB mask* and its macDSMEACT table, which was modified (compared with the one used in DSME) in order to contain these new data fields. The new GTS assignment is communicated through the beacon to all the other nodes in the network that are compliant with D-DSME. Finally, the node sends a notification message M_{not} broadcast to all the other nodes, indicating the newly allocated GTSs, in order to mantain compatibility with the nodes compliant with the standard DSME protocol. M_{not} is sent only if the GTS is allocated to one flow for the first time, while no notification is needed when it is shared multiple times among several flows. This choice is to prevent the nodes that are not compliant with D-DSME from detecting multiple allocation events.

The Shareable D-DSME allocation procedure is up to the *PAN-Coordinator*. The nodes requesting *shGTS* allocation have to wait for a response from the *PAN-Coordinator*, otherwise they cannot use any shGTS. The node that requests a shGTS, after a timeout with no response from the *PAN-Coordinator*, will retransmit the allocation request. During the allocation phase messages can be lost, i.e., the allocation request or the allocation response can be lost. In the first case, as the node requesting the shGTS allocation does not receive any response, it will retransmit the allocation request. Even in the case that the allocation response is lost, the node requesting the shGTS allocation will retransmit the allocation request. In this case the *PAN-Coordinator* replies to the node with the same allocation response transmitted following the first request. The process will repeat until either a reply (positive or negative) is received by the node or a timeout elapses. Hence, termination is always guaranteed. Conversely, if the *PAN-Coordinator* replies to a shGTS request and the node is not able to receive the reply or to transmit other requests (e.g., due to a temporary channel unavailability), the shGTS allocation may result in an inconsistent state. In fact, the shGTS is allocated, but the node cannot use it. However, the inconsistent state has a limited duration, as the node that is not able to communicate will request again a shGTS for the same flow after a given sleep interval and the *PAN-Coordinator* will reply with the same shGTS previously assigned.

Transmission on *shGTSs*. The data in the macDSMEACT table are used by each node to know when it can transmit in the assigned *shGTS*. First, the node determines the current *multi-superframe id* ($MSFId$). This is done using the index ($SFId$) of the superframe that is transmitted by the DSME coordinator in the beacon (Section 7.4.2.5 in [1]). The $SFId$ specifies the index of the current superframe in a beacon interval. If each multi-superframe contains n_{SF} superframes, $MSFId = (SFId \mod n_{SF})$.

A node that wants to know if the *shGTS* with id equal to *shGTSId* within the current multi-superframe is assigned to a flow f checks if the following condition is met.

$$(MSFId - shGTSOffset) \mod shGTSInterval = 0 \tag{4}$$

If condition 4 is met, then the *shGTS* with id equal to *shGTSId* is used to transmit the frames of the flow f.

4. Mirror D-DSME Extension

The standard DSME protocol provides two mechanisms that can be used for acknowledgement and retransmissions. In the first mechanism a message is acknowledged in the same GTS, right after the message transmission. The second mechanism is based on the Group ACK option. When the Group ACK option is enabled, two GTSs, namely GACK1 and GACK2, are used to send ACKs. To take advantage of the Group ACK, it is important to assign two GTSs per each flow within a multi-superframe, i.e., one GTS before GACK1 and another GTS between GACK1 and GACK2. This way, the first GTS is used for the frame transmission. If the transmitted frame is not correctly received, i.e., the ACK of that frame is not received in the GACK1 slot, the second allocated GTS,

namely GTSR (i.e., GTS for Retransmission), will be used for the frame retransmission. The outcome of the retransmission will be checked during the GACK2 slot. Consequently, the Group ACK option supports retransmissions using GTSs dedicated to this scope, but this increases the number of GTSs required by a flow within the multi-superframe.

If no Ack is received, the source node must retransmit the message. Retransmission is performed using another dedicated GTS, but this increases the number of GTSs required by a flow within the multi-superframe.

In industrial applications, the messages are typically generated with different periods, so it may happen than for some flow $P_f > T_{MSF}$ and, therefore, some of the GTSs assigned to the flow within the multi-superframe cannot be used for transmission.

The Mirror D-DSME extension introduces *miGTS (Mirror GTS)*, which exploits these unused GTSs for multiple retransmissions of the same message. For instance, in the scenario shown in Figure 3 each multi-superframe is made up of one superframe (i.e., $T_{MSF} = T_{SF}$). In the CFP, 7 flows $[1 \dots 7]$ are transmitted, with period $P_f = 4T_{SF} = 4T_{MSF}$. As $P_f > T_{MSF}$, the GTS assigned to a flow f is used for the transmission only one multi-superframe over four. As a consequence, the GTS can be labelled as *miGTS*. This way, the three unused occurrences of the GTS are used for message retransmission. The *miGTS* allocation procedure performed by the source node is the same as the standard DSME-GTS allocation one described in Section 3.1.

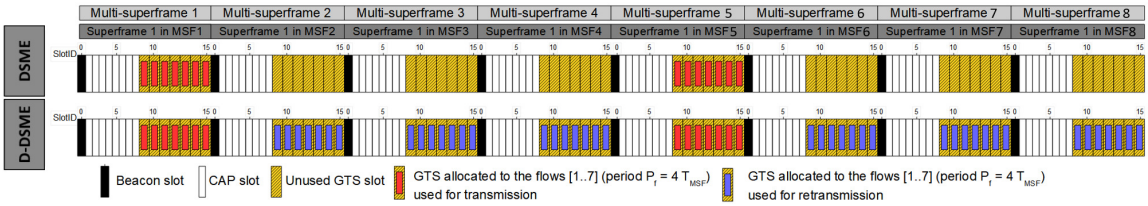

Figure 3. Example using miGTS.

The messages transmitted through *miGTSs* are *unconfirmed*. No Ack reception is supported, so the source node transmits all the configured replicas of the message, thus leaving to the destination node the task of discarding any received duplicate. This means that *miGTS* can reduce the packet loss rate (thus improving reliability), but it cannot guarantee that no message is lost. However, this is an acceptable limitation in industrial applications, as the newest value acquired from a sensor is typically more important than a previous value delivered late.

In typical Industrial WSN applications the network nodes are mains-powered devices; however, in recent industrial applications battery-powered nodes can also be found. In this case, using the Mirror D-DSME extension, multiple retransmissions entail an increase in the energy consumption of the transmitting nodes that is directly proportional to the number of retransmissions allowed for each node, i.e., deltaEnergy = MsgTxEnergy x NumOfRetransmissions. Conversely, it is possible to configure the receiving nodes so that they switch to the sleep mode following the correct reception of a message. Hence, a trade-off between the number of allowed retransmissions (i.e., the reliability) and the energy consumption has to be found. As both these parameters are strictly related to the application requirements and the adopted hardware [30], they have to be tuned during the network design phase.

The *miGTSs* increase the reliability without requiring the allocation of additional GTSs, thus *miGTSs* are useful in networks with nodes that require reliable transmissions. In fact, the adoption of the *miGTS* allows working with several GTSs lower than the one that would be needed using the standard retransmission mechanisms.

5. Performance Assessment

In particular, this section assesses three performance indexes, i.e., scalability, queuing delays for the aperiodic messages transmitted in the CAP, and packet loss ratio. In Section 5.1 we assess

the scalability comparing the number of GTSs required to accommodate all the periodic flows in the case of Shareable DSME-GTS extension enabled and in the case of plain DSME, respectively. For this reason, a network with a star topology is chosen, in which each node generates two periodic flows, with periods multiple of the multi-superframe duration. The simulation is performed increasing the number of nodes. As the assessed metrics, i.e., the number of GTSs required to accommodate all flows, does not depend on the DSME parameters, in this scenario no settings of BO, SO, etc. are needed. Conversely, in the second scenario, discussed in Section 5.2, the average queueing time for the aperiodic messages transmitted in the CAP is assessed. The aim of this assessment is to give some quantitative insights on the improvement in terms of average delays for aperiodic messages when the Shareable DSME-GTS extension is adopted. In this scenario we consider a fixed number of nodes that transmit aperiodic messages and in each simulation run we vary the number of nodes that transmit periodic messages in the GTS. This way the multi-superframe duration varies and, as a consequence, the CAP frequency also varies. Finally, in the third scenario, presented in Section 5.3, the packet loss ratio results of the Mirror D-DSME and the plain DSME protocol, under harsh channel conditions, are compared varying the distance between the end-nodes and the *PAN-Coordinator*.

5.1. Scalability Assessment

We consider that CAP reduction is enabled as this option is very suitable for networks with stringent requirements in terms of delay and reliability, as discussed in [8]. As a consequence, the number of superframes (n_{SF}) within the multi-superframe that is required to support a set of flows (i.e., it contains at least n_{GTS} GTSs) is calculated as:

$$n_{SF} = \begin{cases} 1, & \text{if } n_{GTS} \leq 7 \\ \mathcal{M}\left(1 + \left\lceil \frac{n_{GTS}-7}{15} \right\rceil\right), & \text{otherwise} \end{cases} \quad (5)$$

The $\mathcal{M}(x)$ function is used in Equation (5) because n_{SF}, according to [1], must be a power of 2 (i.e., 2^{MO-SO}). Here we recall that the function $\mathcal{M}(x) = 2^{\lceil log_2(x) \rceil}$ returns the minimum power of 2 that is greater than x (for instance $\mathcal{M}(5) = 8$).

The number of GTSs required (n_{GTS_req}) within the multi-superframe to schedule all the network flows without reaching the network saturation must be calculated in order to assess the scalability.

With the CAP reduction enabled, the number of available GTSs (n_{GTS_avail}) as a function of the number of superframes (n_{SF}) within the multi-superframe is calculated as

$$n_{GTS\_avail} = 7 + [(n_{SF} - 1) \times 15] \quad (6)$$

The network saturation point represents the maximum number of nodes for which n_{GTS_req} is less than or equal to n_{GTS_avail}, under the assumption that all the nodes generate the same flows.

Algorithm 1 (depicted below) calculates the minimum number of superframes (n_{SF}) that contains the GTSs (n_{GTS_req}) required to schedule all the flows. When the algorithm starts, a variable named i, i.e., the value of $MO - SO$, which determines a "test" number of superframe within a multi-superframe ($n_{SF\_test} = 2^{MO-SO}$), is initialized to 0. Moreover, the number of required GTSs is inizialized to 0.

Depending on the flow period, the allocation procedure assigns one or multiple GTSs (shared or not shared) to each flow. Multiple GTSs will be allocated if the flow period is shorter than the multi-superframe length, otherwise only one GTS will be allocated. Firstly, the boolean variable *shared* is initialized to $false$. After that, for each GTS s the algorithm tests if Equations (1) and (2) are met. In such a case, the GTS s is assigned to the flow f and the flow f is inserted in F_s, i.e., in the set of flows that were assigned the GTS s. Furthermore, the variable *shared* is set to true. If no GTS can be assigned to the flow f, new GTSs (one or multiple) are required. Consequently, the number of required GTSs is updated.

When the allocation procedure is completed, the algorithm uses the Equation (5) in order to calculate the number of superframe (n_{SF_req}) required to host n_{GTS_req} GTSs. If $n_{SF\_req} > n_{SF\_test}$,

the duration of the current "test" multi-superframe is not sufficient to schedule the flows. In such a case, the algorithm increases the n_{SF_test} value by increasing the value of i. Next, the allocation steps are repeated iteratively for each flow. Otherwise, if $n_{SF\_req} \leq n_{SF\_test}$, the algorithm ends. The final value of n_{SF_test} provides the MSF duration. The number of "free" GTSs can be calculated as a difference between the n_{GTS_avail} (according to Equation (6)) and the n_{GTS_req}.

Algorithm 1: Algorithm used to assess the network scalability.

```
for i=0; i<15; i++ do
    n_SF_test ← 2^i
    n_GTS_req ← 0
    for all flow f do
        shared ← false
        for all GTS s do
            if Equation 1 and (2) are met then
                insert f to F_s
                shared ← true
                break
            end if
        end for
        if not shared then
            n_GTS_req ← n_GTS_req + ceil(T_MSF / P_f)
        end if
    end for
    n_SF_req ← n_SF as in Equation (5) with n_GTS = n_GTS_req
    if n_SF_req < n_SF_test then
        return n_SF_test
    end if
end for
```

To assess the scalability, a network with star topology was simulated using Algorithm 1 implemented from scratch in Python. Each node periodically transmits to the *PAN-Coordinator* two kinds of real-time messages (f_1 and f_2) with periods equal to $4T_{SF}$ and $8T_{SF}$, respectively. In this assessment no retransmissions and errors are considered, as the aim is to compare the scalability of the standard DSME protocol with and without the D-DSME extensions. The scalability is assessed increasing the number of nodes in the network. The number of GTSs (hence, the number of superframes) within the multi-superframe that are required to schedule all network flows without reaching the network saturation was calculated according to Algorithm 1. The analytical results are shown in Figure 4.

The D-DSME scenario requires a lower number of slots (dark lines in Figure 4) compared with the standard DSME (light lines in Figure 4). This entails a higher scalability even in terms of the number of flows that can be supported. In fact, in the case of 25 nodes (in Figure 4), the number of available GTSs with 4 superframe is equal to 52. Using the standard DSME protocol 50 out of 52 GTSs are required. Conversely, using the D-DSME only 38 out of 52 GTSs are required, thus there are 14 GTSs that can be assigned to additional flows (12 GTSs more than the standard DSME protocol).

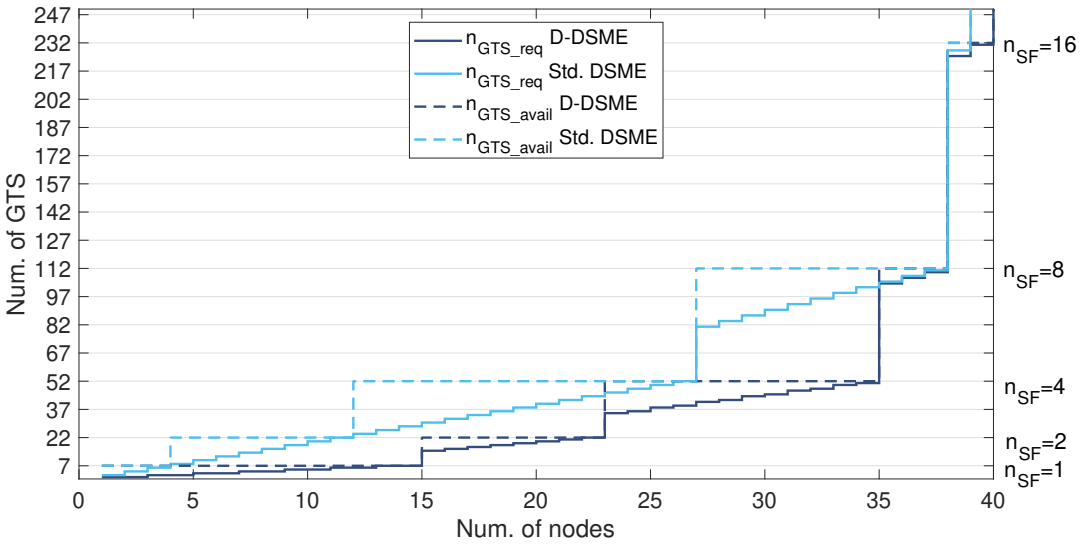

Figure 4. Superframes required to accommodate all flows.

In both scenarios, when there are more than 39 nodes for the DSME and 40 nodes for the D-DSME, the network reaches saturation (i.e., the number of required GTSs exceeds the number of available GTSs, as it was explained at the beginning of this subsection. Hence, in the worst-case, i.e., before reaching the network saturation, the D-DSME scalability performance are the same than the standard DSME ones.

The results show that when the multi-superframe period is lower than the period of the flows f_2, in this scenario, the D-DSME provides a higher scalability than the standard DSME protocol. In this scenario, with $n_{SF} = 4$, the improvement of the number of nodes that can be supported is 30% higher than in the standard DSME protocol.

D-DSME, compared to the standard DSME protocol, reduces the number of required GTSs, with the consequence of a higher number of available GTSs, thus providing two advantages. First, with a fixed number of superframes within the multi-superframe, the D-DSME provides more room for new periodic flows. Second, as the number of superframes within the multi-superframe can be reduced to provide a shorter multi-superframe, the CAP frequency increases and the average delays for aperiodic non real-time messages are shortened.

5.2. Queuing Delay Assessment

To measure the queuing delay ($\mathcal{D}$) of the aperiodic messages transmitted in the CAP, i.e., the time spent by aperiodic messages in the queue while waiting for transmission, we run simulations using the OMNeT++ framework. The IEEE 802.15.4 physical layer model adopted is the one provided by the INETMANET libraries, while the MAC layer model was developed from scratch. $\mathcal{D}$ was measured both with and without enabling the D-DSME extensions. The simulations were run with the parameters that are listed in Table 2.

The simulated network includes: 25 aperiodic nodes (ANs) that transmit aperiodic messages of 59B during the CAP; $n_{nd} \in [1, \ldots, 34]$ real-times nodes (RTNs) that transmit two kinds of real-time messages (f_1 and f_2), respectively, during the CFP; one *PAN-Coordinator* that receives the messages transmitted by the end nodes. The aperiodic messages are generated by the ANs with a random period, following an exponential distribution with mean 31.2 ms. The periodic messages belonging to f_1 and f_2 flows are generated by the RTNs with periods equal to $P_{f_1} = 4T_{SF}$ and $P_{f_2} = 8T_{SF}$, respectively.

Table 2. Simulation parameters.

Parameter	Value
Superframe Duration (T_{SF})	61.44 ms
GroupAck	Disabled
CAP reduction	Enabled
macMinBE	5
maxMaxBE	5
macMaxCSMABackoffs	4
macMaxQueueLength	590 bytes

As the number n_{nd} of RTNs varies for each simulation, n_{SF} changes according to Equation (5) (and so, the MSF duration and the CAP frequency). Therefore, the queuing delay $\mathcal{D}$ depends on n_{nd} and varies for each simulation. The measured average $\mathcal{D}$ values are shown in Table 3. Every simulation was repeated multiple times, so as to obtain confidence intervals calculated at 95% in the order of tens of milliseconds.

Table 3. Simulation results.

Min..Max Supported RTNs (n_{nd})		n_{SF}	Avg. Queueing
Std. DSME	**D-DSME**		**Delay $\mathcal{D}$**
1...3	1...4	1	$0.32\,\text{s} \pm 0.005\,\text{s}$
4...11	5...14	2	$0.79\,\text{s} \pm 0.01\,\text{s}$
12...26	15...34	4	$1.60\,\text{s} \pm 0.02\,\text{s}$

Table 3 shows that in some cases the queueing delay $\mathcal{D}$ can be lower for D-DSME than for DSME. For instance, using a network with $n_{nd} = 13$ RTNs, the required SFs are $n_{SF} = 4$ for DSME and only $n_{SF} = 2$ for D-DSME. In such a case, D-DSME outperforms DSME (the measured $\mathcal{D}$ values are respectively 0.79s and 1.60s). This happens because D-DSME allows using a shorter multi-superframe, thus increasing the CAP frequency.

5.3. Packet Loss Ratio Assessment

To evaluate the improvements introduced by the Mirror D-DSME extension a comparative assessment was performed using the OMNeT++ framework, with the simulation parameters shown in Table 2 (i.e., the same ones used in Section 5.2).

The assessed metric is the Packet Loss Ratio (PLR), defined as the percentage of lost messages over the number of messages generated at the application level, i.e.,

$$PLR = \left(1 - \frac{ReceivedMessages}{GenMessages}\right) \times 100 \tag{7}$$

where *ReceivedMessages* is the number of messages received at the sink application layer and *GenMessages* is the number of messages generated at the source application layer.

The simulated scenario includes one *PAN-Coordinator* that receives real-time messages from 7 end nodes. Each End node generates a flow f made up of 59-byte messages with a period $P_f = 4T_{SF}$.

To assess the reliability under different channel conditions, the Log-normal shadowing channel model with $\alpha = 3.44$ and $\sigma = 7$ was adopted, as these values provide a very harsh channel environment, similar to a realistic industrial context [31]. Results were obtained increasing the distance of the nodes from the *PAN-Coordinator* that acts as the sink.

Two configurations (one for DSME and one for D-DSME) were considered. In the DSME configuration, the messages are confirmed and the Group ACK is enabled, while in the D-DSME configuration the messages are unconfirmed and the Group ACK is disabled. The simulation parameters (excluding the above discussed Group ACK option) are shown in Table 2. In D-DSME

each multi-superframe contains only one superframe, therefore $T_{SF} = T_{MSF}$. As the flow period is $P_f = 4T_{SF} > T_{MSF}$, the *miGTS* in the first multi-superframe can be used to transmit the message. The *miGTSs* in the following three multi-superframes are used for retransmissions, and so on.

In the D-DSME configuration, the same (unconfirmed) message can be transmitted (or retransmitted) four times as a maximum. Conversely, with the standard DSME approach the (confirmed) message is transmitted twice as a maximum.

Figure 5 shows the measured PLR values. They are significantly lower when the Mirror D-DSME extension is adopted. This is a reasonable result, as in the D-DSME scenario an end node has more chances of transmitting the same message than with the standard DSME protocol.

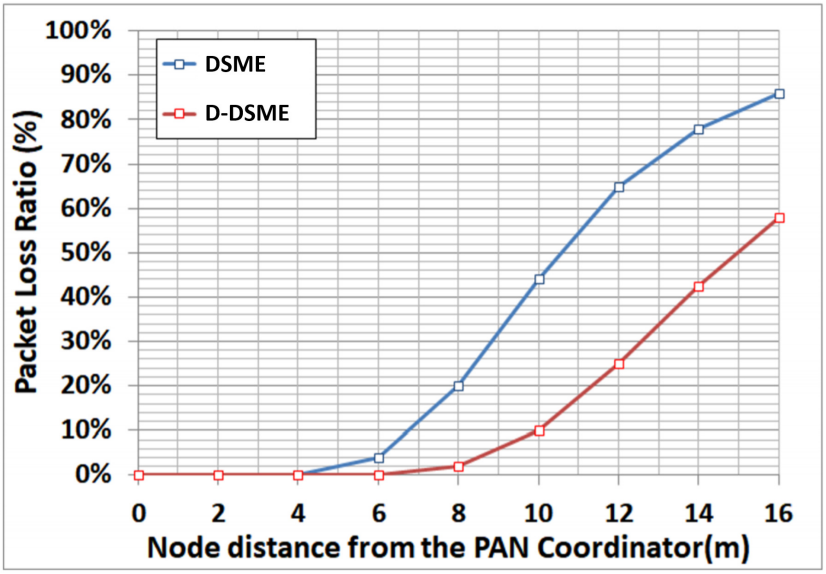

Figure 5. Packet Loss Ratio results.

In particular, the lower the GTS use, the higher are the advantages in terms of PLR that the Mirror D-DSME extension provides. Clearly, if no flow has period $P_f > T_{MSF}$ there will be no unused slots. In such a case, the Mirror D-DSME extension does not improve the standard DSME protocol.

6. Implementation of D-DSME on Real Devices

An implementation of the proposed D-DSME extensions was developed on boards equipped with the PIC24FJ256GB108 microcontroller and the MRF24J40MA transceiver, both produced by Microchip Tech. The communication between the transceiver and the microcontroller takes place via the Serial Peripheral Interface (SPI).

The implementation does not require any hardware modifications. The D-DSME extensions were implemented from scratch and the classic IEEE 802.15.4-2006 MAC layer, implemented in the transceiver, was disabled. The considered scenario is a network with a tree topology that includes one *PAN-Coordinator*, one coordinator, and three end nodes. All the end nodes generate two flows, f_1 and f_2, with period equal to $2T_{SF}$. The messages (with a 8-bytes payload) are transmitted to the coordinator that sends them to the *PAN-Coordinator*. The end-to-end delays are acquired using the GPIOs integrated in the MCUs of the source and destination nodes. The configuration parameters are the same as the ones in Table 2. Table 4 shows that the measured end-to-end delays are similar to the ones obtained in simulation. The time differences, which are always between 1409 µs and 1949 µs, are due to the processing delays that were not considered in the simulation. These times are acceptable if compared with the obtained delays. In the considered scenario, no packet was lost.

Table 4. Comparative end-to-end delays (μs).

Node ID	Flow f_1	Flow f_2	Flow f_1	Flow f_2	Flow f_1	Flow f_2
	Simulated		Experimental		Difference	
1	42,976	104,416	44,429	106,365	1453	1949
2	46,816	108,256	48,249	109,975	1433	1719
3	50,656	112,096	52,065	113,750	1409	1654

7. Conclusions

This work proposed the D-DSME extensions, which improve the scalability and reliability of the DSME protocol. In particular, the Shareable D-DSME extension allows GTSs to be shared between multiple flows, while the Mirror D-DSME extension allows for multiple retransmissions on the same GTS. The strong point of the Shareable D-DSME extension is that it is beneficial to both periodic and aperiodic flows. The first benefit from the slot sharing among multiple flows is that it leaves room for accommodating in a multi-superframe of the same length a larger number of periodic flows than DSME. Aperiodic flows, instead, benefit from the reduced multi-superframe length that, in case of CAP reduction enabled, determines shorter queuing delays for the aperiodic messages waiting for transmission within the CAP. The strong point of the Mirror D-DSME is that it exploits the same GTS for both the message transmission and the relevant retransmissions, thus improving reliability without requiring additional slots. As shown in the paper, both the proposed extensions can be implemented on commercial devices without hardware modifications. Currently, approaches to optimize the Shareable GTS assignment algorithm are under study and will be addressed in future work. Moreover, future works will address the case for Software Defined Networking architectures [32,33] as a mean to make scheduling and GTS allocation in Shareable D-DSME even more flexible.

Author Contributions: Conceptualization, methodology, investigation, writing–original draft preparation, and writing–review and editing: F.B., M.C., L.L., L.L.B., G.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011). *IEEE Standard for Low-Rate Wireless Networks*; IEEE Standards Association: Piscataway, NJ, USA, 2016.
2. Leonardi, L.; Patti, G.; Battaglia, F.; Lo Bello, L. Simulative assessments of the IEEE 802.15.4 CSMA/CA with Priority Channel Access in Structural Health Monitoring scenarios. In Proceedings of the IEEE 15th International Conference on Industrial Informatics (INDIN 2017), Emden, Germany, 24–26 July 2017.
3. IEEE Std 802.15.4e-2012. *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC Sublayer*; IEEE Standards Association: Piscataway, NJ, USA, 2012.
4. Alderisi, G.; Patti, G.; Mirabella, O.; Lo Bello, L. Simulative assessments of the IEEE 802.15.4e DSME and TSCH in realistic process automation scenarios. In Proceedings of the 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), Cambridge, UK, 22–24 July 2015; pp. 948–955. [CrossRef]
5. Patti, G.; Alderisi, G.; Lo Bello, L. Introducing multi-level communication in the IEEE 802.15.4e protocol: The MultiChannel-LLDN. In Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA), Barcelona, Spain, 16–19 September 2014; pp. 1–8. [CrossRef]
6. Meyer, F.; Mantilla-González, I.; Kauer, F.; Turau, V. Performance Analysis of the Slot Allocation Handshake in IEEE 802.15.4 DSME. In *Ad-Hoc, Mobile, and Wireless Networks*; Palattella, M.R., Scanzio, S., Coleri Ergen, S., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 102–117.
7. De Guglielmo, D.; Brienza, S.; Anastasi, G. IEEE 802.15.4e: A Survey. *Comput. Commun.* **2016**, *88*. [CrossRef]

8. Kurunathan, H.; Severino, R.; Koubaa, A.; Tovar, E. IEEE 802.15.4e in a Nutshell: Survey and Performance Evaluation. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 1989–2010. [CrossRef]
9. Kauer, F.; Köstler, M.; Lübkert, T.; Turau, V. OpenDSME—A portable framework for reliable wireless sensor and actuator networks. In Proceedings of the 2017 International Conference on Networked Systems (NetSys), Gottingen, Germany, 13–16 March 2017.
10. Dutta, P.; Hui, J.; Jeong, J.; Kim, S.; Sharp, C.; Taneja, J.; Tolle, G.; Whitehouse, K.; Culler, D. Trio: Enabling sustainable and scalable outdoor wireless sensor network deployments. In Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN), Nashville, TN, USA, 19–21 April 2006; pp. 407–415. [CrossRef]
11. Suhonen, J.; Kohvakka, M.; Kaseva, V.; Hämäläinen, T.; Hännikäinen, M. *Low-Power Wireless Sensor Networks. Protocols, Services and Applications*; Springer Science & Business Media: Berlin, Germany, 2012. [CrossRef]
12. Iannizzotto, G.; La Rosa, F.; Costanzo, C.; Lanzafame, P. A multimodal perceptual user interface for collaborative environments. In Proceedings of the International Conference on Image Analysis and Processing, Cagliari, Italy, 6–8 September 2005; pp. 115–122.
13. Taneja, M. A framework to support real-time applications over IEEE802.15.4 DSME. In Proceedings of the 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Singapore, 7–9 April 2015; pp. 1–6. [CrossRef]
14. Battaglia, F.; Collotta, M.; Leonardi, L.; Lo Bello, L.; Patti, G. A scalable approach for periodic traffic scheduling in IEEE 802.15.4-DSME networks. In Proceedings of the IEEE 17th International Conference on Industrial Informatics (INDIN 2019), Helsinki, Finland, 22–25 July 2019.
15. De Guglielmo, D.; Seghetti, A.; Anastasi, G.; Conti, M. A performance analysis of the network formation process in IEEE 802.15.4e TSCH wireless sensor/actuator networks. In Proceedings of the 2014 IEEE Symposium on Computers and Communications (ISCC), Funchal, Portugal, 23–26 June 2014; pp. 1–6. [CrossRef]
16. Patti, G.; Lo Bello, L. A Priority-Aware Multichannel Adaptive Framework for the IEEE 802.15.4e-LLDN. *IEEE Trans. Ind. Electron.* **2016**, *63*, 6360–6370. [CrossRef]
17. Toscano, E.; Lo Bello, L. A multichannel approach to avoid beacon collisions in IEEE 802.15.4 cluster-tree industrial networks. In Proceedings of the 2009 IEEE Conference on Emerging Technologies Factory Automation, Mallorca, Spain, 22–25 September 2009; pp. 1–9. [CrossRef]
18. Jeong, W.-C.; Lee, J. Performance evaluation of IEEE 802.15.4e DSME MAC protocol for wireless sensor networks. In Proceedings of the 2012 The First IEEE Workshop on Enabling Technologies for Smartphone and Internet of Things (ETSIoT), Seoul, Korea, 18 June 2012; pp. 7–12. [CrossRef]
19. Lee, J.; Jeong, W. Performance analysis of IEEE 802.15.4e DSME MAC protocol under WLAN interference. In Proceedings of the 2012 International Conference on ICT Convergence (ICTC), Jeju Island, Korea, 15–17 October 2012; pp. 741–746. [CrossRef]
20. Sahoo, P.K.; Pattanaik, S.R.; Wu, S.L. A reliable data transmission model for IEEE 802.15. 4e enabled wireless sensor network under wifi interference. *Sensors* **2017**, *17*, 1320. [CrossRef] [PubMed]
21. Alsudany, S.K.; Boussakta, S.; Johnston, M. Enhancements of IEEE802.15.4e DSME Model of Wireless Sensor Networks. In Proceedings of the 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN), Prague, Czech Republic, 3–6 July 2018; pp. 426–431. [CrossRef]
22. Liu, X.; Li, X.; Su, S.; Fan, Z.; Wang, G. Enhanced Fast Association for 802.15.4e-2012 DSME MAC Protocol. In *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering*; Atlantis Press: Paris, France, 2013. [CrossRef]
23. Sahoo, P.K.; Pattanaik, S.R.; Wu, S.-L. A Novel IEEE 802.15.4e DSME MAC for Wireless Sensor Networks. *Sensors* **2017**, *17*, 168. [CrossRef] [PubMed]
24. Kurunathan, H.; Severino, R.; Koubaa, A.; Tovar, E. RPL over DSME: A Link-Asymmetric QoS-Qware Communication Protocol Stack for Scalable IoT Networks. *Technical Report*, 2018. Available Online: https://www.cister.isep.ipp.pt/docs/rpl_over_dsme__a_technical_report/1347/view.pdf (accessed on 12 December 2019).
25. Kauer, F.; Köstler, M.; Turau, V. Reliable Wireless Multi-Hop Networks with Decentralized Slot Management: An Analysis of IEEE 802.15.4 DSME. *arXiv* **2018**, arXiv:1806.10521.

26. Kurunathan, H.; Severino, R.; Koubaa, A.; Tovar, E. DynaMO - Dynamically tuning DSME Networks. In Proceedings of the 17th International Workshop on Real-Time Networks (RTN2019), Stuttgart, Germany, 9 July 2019.
27. Kurunathan, H.; Severino, R.; Koubaa, A.; Tovar, E. DynaMO -Dynamic Multisuperframe Tuning for Adaptive IEEE 802.15.4e DSME Networks. *IEEE Access* **2019**, *7*, 122522–122535. [CrossRef]
28. Lee, Y.S.; Chung, S.H. An Efficient Distributed Scheduling Algorithm for Mobility Support in IEEE 802.15.4e DSME-Based Industrial Wireless Sensor Networks. *Int. J. Distrib. Sens. Netw.* **2016**, *12*, 9837625. [CrossRef]
29. Kauer, F.; Köstler, M.; Turau, V. openDSME: Reliable Time-Slotted Multi-Hop Communication for IEEE 802.15.4. In *Recent Advances in Network Simulation: The OMNeT++ Environment and Its Ecosystem*; Springer International Publishing: New York, NY, USA, 2019; pp. 451–467.
30. Juc, I.; Alphand, O.; Guizzetti, R.; Favre, M.; Duda, A. Energy consumption and performance of IEEE 802.15.4e TSCH and DSME. In Proceedings of the 2016 IEEE Wireless Communications and Networking Conference, Doha, Qatar, 3–6 April 2016; pp. 1–7. [CrossRef]
31. Tanghe, E.; Joseph, W.; Verloock, L.; Martens, L.; Capoen, H.; Van Herwegen, K.; Vantomme, W. The Industrial Indoor Channel: Large-Scale and Temporal Fading at 900, 2400, and 5200 MHz. *IEEE Trans. Wirel. Commun.* **2008**, *7*, 2740–2751. [CrossRef]
32. Lo Bello, L.; Lombardo, A.; Milardo, S.; Patti, G.; Reno, M. Software-Defined Networking for Dynamic Control of Mobile Industrial Wireless Sensor Networks. In Proceedings of the IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Turin, Italy, 4–7 September 2018.
33. Leonardi, L.; Ashjaei, M.; Fotouhi, H.; Lo Bello, L. A Proposal Towards Software-Defined Management of Heterogeneous Virtualized Industrial Networks. In Proceedings of the IEEE 17th International Conference on Industrial Informatics (INDIN 2019), Helsinki, Finland, 22–25 July 2019.