Computer Networks 213 (2022) 109114

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

PSM-DMO: Power Save Mode and Discontinuous BLE Mesh Operation

Á. Hernández-Solana [a,*], D. Perez-Diaz-de-Cerio [b], M. García-Lozano [b], A. Valdovinos [a], J.L. Valenzuela [b]

[a] *Aragon Institute for Engineering Research (I3A), University of Zaragoza, Zaragoza 50018, Spain*
[b] *Signal Theory and Communications Department, Universitat Politècnica de Catalunya, Esteve Terrades 7, Castelldefels 08860, Spain*

ARTICLE INFO

Keywords:
Bluetooth
BLE
Energy
Power saving
Bluetooth mesh
Wireless mesh networks

ABSTRACT

The Bluetooth Low Energy (BLE) mesh profile, standardized by the Bluetooth Special Interest Group (SIG), has an increasing interest in IoT solutions. However, the standard assumes that relay and friend nodes should be continuously scanning the channel awaiting any incoming transmissions. This could be very inefficient in terms of energy consumption, particularly in application scenarios where the backbone of the mesh network cannot be powered and traffic is infrequent. Hence, we present a novel strategy, named PSM-DMO, that minimizes the scan periods and thus, significantly reduces the overall energy consumption of the mesh network. PSM-DMO is defined as a new and optional feature for the currently published BLE mesh specifications, coexists with the standard operation, and is implemented without modifying the core of the specification. The proposal, that ensures the reliability of the mesh operation, can be used in BLE sensor networks that can tolerate a certain transmission delay. PSM-DMO replaces the continuous scan by a periodic but asynchronous polling process whereby the relay and sink nodes interrogate their neighbors about the existence of data to receive or to retransmit through the network. Nodes only go into scan mode during the period of time the mesh network will be involved in the transmission and dissemination. This period is estimated by the node which is the source of data, it is announced to its neighbors and it is propagated consecutively by all the relay nodes until the destination. PSM-DMO allows a theoretical reduction in the energy consumption of relay nodes up to 99.24 %.

1. Introduction

Wireless mesh networking has been proposed as a suitable solution in sensor networks for a wide range of application fields [1–3]: agriculture, farm and cattle control, mountain hydrology measurements, national park monitoring, mobility, smart cities, and in the industry, in the often called Industrial Internet of the Things (IIoT) [4,5]. Meshed architectures allow to improve the control, monitoring, and automation in terms of robustness, reliability, security, latency, and jitter. The main requirements of the applications are features such as resilience, reliability, low infrastructure costs [6–8], and in many cases, ultra-long battery life, preferably measured in years [9,10]. We focus on those applications in which access to energy resources is extremely limited and the operational costs due to battery handling would be prohibitive. In these cases, the devices need to work autonomously. Also, such applications usually require low throughput. Their data generation can be sparse and do not have low-latency requirements. For instance, this is the case of beer fermentation monitoring [11], requiring a sampling rate of six hours, smart agriculture applications that need to send one message per hour [12], smart health [13] or the smart cities and mobility field, that would just need to transmit 10 detections per day [14]. In this context, we explore the potentiality of solutions based on Bluetooth Low Energy (BLE) mesh.

In the last years, BLE has gained a dominant position for IoT and other sensor-based applications thanks to its simplicity, low-power consumption, low-cost, and robustness. Really, one of the main advantages of Bluetooth over its competitors is its widespread implantation in the market. The new mesh specification [15] based on Bluetooth Low Energy [16] and developed by the Bluetooth SIG has increased the interest and position of BLE as an enabler of IoT and IIoT, compared with other technological proposals supporting mesh networking, such as Zigbee, WirelessHart, 6LoWPAN or Thread, all of them based on the IEEE 802.15.4 standard family. Nevertheless, contrary to other mesh networks or protocols (including ZigBee, Thread, Z-Wave, WiFi) that use routing techniques, BLE mesh only considers managed flooding. This strategy provides a simple and robust transmission mechanism but

* Corresponding author.
E-mail address: anhersol@unizar.es (Á. Hernández-Solana).

https://doi.org/10.1016/j.comnet.2022.109114
Received 16 December 2021; Received in revised form 7 April 2022; Accepted 14 June 2022
Available online 23 June 2022

requires that nodes scan the channel continuously. The BLE mesh standard defines: “A device supporting only the advertising bearer should perform passive scan with a duty cycle as close to 100 percent as possible to avoid missing any incoming mesh messages or provisioning Packet Data Units (PDUs)” [15]. Continuous scan implies that BLE loses part of its low-energy principles when used in a mesh network, particularly in application scenarios where transmissions are infrequent. Therefore, this paper presents a novel strategy to reduce the overall consumption, named Power Save Mode and Discontinuous BLE Mesh Operation (PSM-DMO).

The proposal, adapted to the BLE mesh specification, allows all devices/nodes of the mesh network to go into scan mode only when there is a sensor/device with data to transmit and flooding is required. It replaces the continuous scan by a periodic but asynchronous polling process whereby the nodes interrogate their neighbors about the existence of data to receive or to retransmit through the network. Relay and sink nodes remain by default in sleep mode between polls to save energy and they only go into scan mode during the period of time the mesh operation reactivates. This period is estimated by the node which is the source of data. It is announced to its neighbors and it is propagated consecutively by all the relay nodes until the destination.

This proposal clearly differs from the concept of Bluetooth Mesh Low Power Node (BM-LPN) [17]. LPNs, that are located at the end of the network (they cannot act as relays), can be inactive for a long period if their transmissions are infrequent and they have no low-delay requirements. For instance, sensors in scenarios like the ones described in [11–14] . However, they should be connected to a friend node that must remain permanently active. Some approaches for the optimization of the friendship mechanism have recently introduced in [17]. Nevertheless, LPNs do not prevent that relay and friend nodes must remain constantly listening for transmissions, which implies high energy consumption. Note that a real device consumes around 5 mA while scanning [18,19]. This means that a relay powered by a 10,000 mAh battery would have a battery life of under three months. That is, due to continuous scan, mesh BLE loses part of the BLE low-energy principles. This issue is remarked in [20], a remarkably interesting study where we can find a model of BLE mesh energy consumption based on measurements performed on a real hardware platform using the nRF51422 chipset from Nordic Semiconductor [21]. Proposal of [20] is useful to evaluate the energy consumption of the current state of the standard [15]. However, conclusions of [20] state that “Bluetooth Mesh cannot be considered as a general-purpose Internet of the Things (IoT) technology because its application is limited to the scenarios where its backbone can be appropriately powered”. Contrary to [20], we see in [22] that BLE mesh can also be interesting for this kind of applications limiting the scan cycles.

PSM-DMO differs from the BMADS proposal previously presented by the authors in [22] for the same purpose. BMADS is based on dynamic scan cycles and sending of a new control message sequence that puts the nodes into a continuous scan. PSM-DMO replaces the requirement of periodic scan cycles with a periodic polling scheme. As we will see, the energy consumption linked to the periodic polling is effectively lower than scan cycles proposed in BMADS.

Concerning the state of the art, works focusing on energy optimization in BLE mesh are scarce. Although energy saving is rapidly becoming an interesting and hot topic in mesh networks, in most cases, authors do not focus on energy optimization but on other issues: energy consumption estimation, application of BLE mesh in new areas, or exploring mesh alternatives, that clearly differ from the standardized managed flooding mechanism. For instance, in [23], authors survey several deployed applications, the problems to adopt BLE in new areas, and current academic and industrial solutions that expand the capabilities of BLE, as can be the new mesh profile. However, energy optimization is not discussed. In [24], authors survey works done on Bluetooth multi-hop networks, and include the BLE mesh as one of the enabler technologies among other BLE possibilities. Nevertheless, the survey mostly emphasizes energy saving in solutions that change the routing or use a scatternet topology (connection-oriented), while energy savings in BLE mesh are not addressed. In [25], authors analyze and compare, using power consumption as one of the metrics, flooding using the Trickle algorithm [26], and a connection-oriented networking solution called Fruitymesh [27]. They conclude that the optimal mesh approach depends on the application. Focusing on energy saving, the Trickle algorithm as well as the most recent Drypp algorithm presented in [28] help to reduce the amount of redundant network traffic by adapting transmission rates to network density. This reduction has a clearly impact on energy consumption. However, Trickle and Drypp continue to use a continuous scan approach. Thus, the impact over the energy consumption is considerably lower than the proposal presented here, which allows all nodes of the mesh network to go into scan mode only when a device needs to transmit data.

Concerning the FruityMesh solution, like Greenlink [29] or the work discussed in [30], it really is a connection-oriented solution. Thus, the energy savings are moderate and the approach greatly differs from the current non-connection oriented BLE mesh approach. FruityMesh is based on neighbor-only routing, where no routing tables are stored but a connection is established between two nodes and kept open. Greenlink [29], which is a new technology of scatternet formation for BLE, and the work presented in [30] are very interesting approaches. Nevertheless, they really focus on minimizing the number of central/relay nodes, same as the Minimum Relay Tree presented in [31]. Greenlink reduces energy consumption a 50% compared with the standard deployment. Compared to Greenlink, the Minimum Relay Tree requires a previous analysis of the network and the manual selection of which nodes should act as relays. If the topology changes, the network should be reconfigured again manually. Low Power Listening mechanisms like the ones listed for other wireless sensor technologies in [32] can also be explored as another strategy to reduce consumption. In [32], the paper presents three main categories: scheduled, protocols with Common Active Periods, and asynchronous MAC. However, from our point of view, the use of these proposals or a store-and-forward solution implies a drastic change to the bearer, network, and transport layers of the BLE mesh standard. So, most of them cannot be directly used in BLE mesh. Other proposals, like RFC7668 [33], which adapts IPv6 to BLE, or like [34], which presents a novel architecture for IPv6 over BLE mesh networks, are far from the proposal of this work. The first is an adaptation based on connection mode and BLE mesh is based on the transmission of broadcast messages. The second uses continuous scan mode, so the energy consumption is still high.

Looking at the orientation of the works found in the literature, the differences of the proposal presented here are clear and significant. Realize that the main motivation of the PSM-DMO proposal, like the BMADS scheme previously proposed by the authors in [22], is that it does not modify the core of the standard as other approaches do. We consider that is one of the main requirements so that the industry can adopt it more easily and quickly. In addition, the proposal must be based on actual devices and measurements to be a realistic approach. Moreover, it should allow large energy savings, particularly in IoT and IIoT application scenarios where traffic is sporadic or infrequent, even when the size of the networks (number of hops) is high. The goal is to achieve a substantial multiplication of the battery life time of the devices. We will show that with few adjustments, the actual BLE mesh can be considered as a general-purpose of IoT and IIoT in many scenarios, allowing large energy savings up to 99.24%. That is, it is up to 8 times better than the BMADS proposal and much more than the 50% claimed by solutions like Greenlink, knowing that the main principles of the Greenlink approach are different and thus, the comparison unfair. In any case, we recall that the proposal must be defined as an optional operation mode, coexisting with the standard mesh operation and eligible when nodes operate in scenarios where energy saving is significant compared with the standard operation. It has been designed as a new feature that can be applied directly to the currently published mesh profile unlike [27,30], etc.

The paper is organized as follows. First, we review the basics of Bluetooth mesh with its main parameters, protocol stack, and frame definitions. Then, in Section 3, we present the points that define our proposal. Later, in Section 4, we analyze the energy consumption of our proposal in comparison with the standard behavior of the specifications. Section 5 discusses the principles of parameter selection and we present the performance of the proposal under different scenarios, compared with the standard operation and the BMADS proposal. Finally, Section 6 concludes the paper.

2. Overview of Bluetooth Mesh basis and BLE bearers

In this section, we will introduce the key aspects that characterize Bluetooth Mesh to understand timing relationships between transmissions involved in the proposal, encapsulation of data to be transmitted (overheads), and the mechanism used to achieve the reliability of the network.

Devices that are part of a mesh network, specifically devices that can transmit and receive messages, are called "*Nodes*" or *"provisioned devices"*. Additionally, a node may have one of several optional features, giving them special capabilities: *relay, friend, low power, and proxy*. From all of them, the *relay feature* deserves special attention because, as in any mesh network, in BLE mesh, the nodes with this feature can receive and then retransmit mesh messages. Thus, they cooperate dynamically to transport messages across the network, allowing large deployments.

Bluetooth mesh has been designed as a layered architecture with BLE 4.x backward compatibility. The main characteristic of the BLE mesh standard is that it is built on top of the full BLE core stack (physical and link layer), as illustrated in Fig. 1, and uses a managed flooding mechanism instead of routing. This mechanism is very robust. It allows immediate sending of messages, and does not require maintenance of the routes. However, it requires that relay nodes scan the channel continuously.

In the current version, mesh data are transmitted consecutively on primary channels 37, 38, and 39 reserved for all non-connected state communications and using advertising events (see Fig. 2) as a bearer. Really, there are two bearers defined: the advertising bearer for devices that include the mesh profile into their stack and the GATT (Generic Attribute) bearer. GATT bearer is used by devices that do not support natively the Bluetooth Mesh Profile to communicate with a mesh network through a proxy node. We focus on nodes supporting the mesh profile.

The specification states that the advertising bearer shall use *non-connectable* and *non-scannable undirected advertising events* (see bottom of Fig. 2) with some little adaptation. Besides, according to the specifications, all devices supporting only the advertising bearer should be scanning with a duty cycle as close as possible to 100 %.

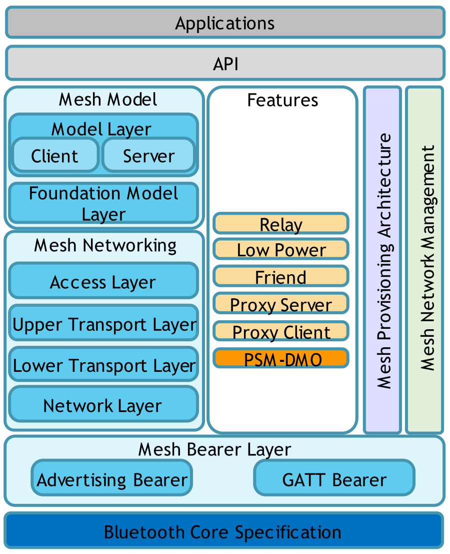


Fig. 1. Mesh BLE stack and required changes to include the new feature.

The standard follows a layered architecture covering all the OSI layers. On top of the mesh stack, an application is implemented. One of the most important key points of BLE mesh is the specification of device behaviors according to a model paradigm (set of configuration states that concerns the node capabilities and behavior within the mesh: features, addresses the node has subscribed to, the security keys, etc.). In such a way, the full BLE mesh stack (see Fig. 1) deals with:

- Model layer/foundation model layer: to define and implement the models and basic functionality of nodes (behaviors, states, messages, and so on) in specific application scenarios.
- Access layer: to define how higher layers (application) can use the more technical layers below.
- Transport layer: to encrypt, decrypt and authenticate application data (at the upper transport layer), and to define how upper transport layer messages are segmented and reassembled into multiple lower transport PDUs.
- Network layer: to define how transport-layer messages are addressed towards one or more elements. That is, it defines the network message format that allows Transport PDUs to be transported by the bearer layer and whether to relay/forward messages accept them for further processing or reject them.
- Bearer layer: to abstract the underlying BLE Core [16] specification towards the layers above through the so-called bearer concept, defined above.

As we stated above, BLE mesh uses a managed flooding technique, supported by two main features:

1) Each message includes a Time To Live field (TTL value included at the network layer) that limits the number of times a message can be relayed and,
2) the nodes maintain a cache of the last received messages to avoid unnecessary retransmissions of previously received messages.

Firstly, BLE mesh preferably uses advertising bearers, based on *non-connectable and non-scannable advertising events*. These events are composed by the transmission of one advertising ADV_NONCONN_IND PDU in sequence using at least one of the three advertising channels (channel 37, 38, and/or 39). The time between the start of two consecutive advertising events (affected by the imposed timing restrictions within the event) is controlled by the *advInterval* parameter ($\geq$20 ms) plus a random variable between 0 and 10 ms.

In addition, the reliability of the mesh network is based on the relay and repetition of the messages according to three different procedures and parameters that could be redundant. One of them is controlled by the model-related layers and the other two by the network layer.

Publish retransmit. This process is controlled at the *model* level. It defines how many replicas of an *access packet* (containing the model messages) are generated and the time between them. This is controlled by the *Publish retransmit count* (P_{RC}) parameter and the publish interval (*pubInt*), which depends on the *Publish retransmit interval steps* (P_{RIS}) parameter (5-bit value). P_{RC} takes values between zero and seven, i.e., a message can be published between one and eight times. *The pubInt* interval equals to (P_{RIS}+1)·50 ms and takes values from 50 to 1600 ms using 50 ms steps.

Network transmit. This process is controlled at the network layer. It defines the number of transmissions of a network PDU generated in a source node and the interval between them. This is controlled by the *Network transmit count* (N_{TC}) parameter and the network interval (*netInt*), which depends on the *Network transmit interval steps* (N_{TIS}) parameter (5-bit value). N_{TC} takes values between zero and seven, so a

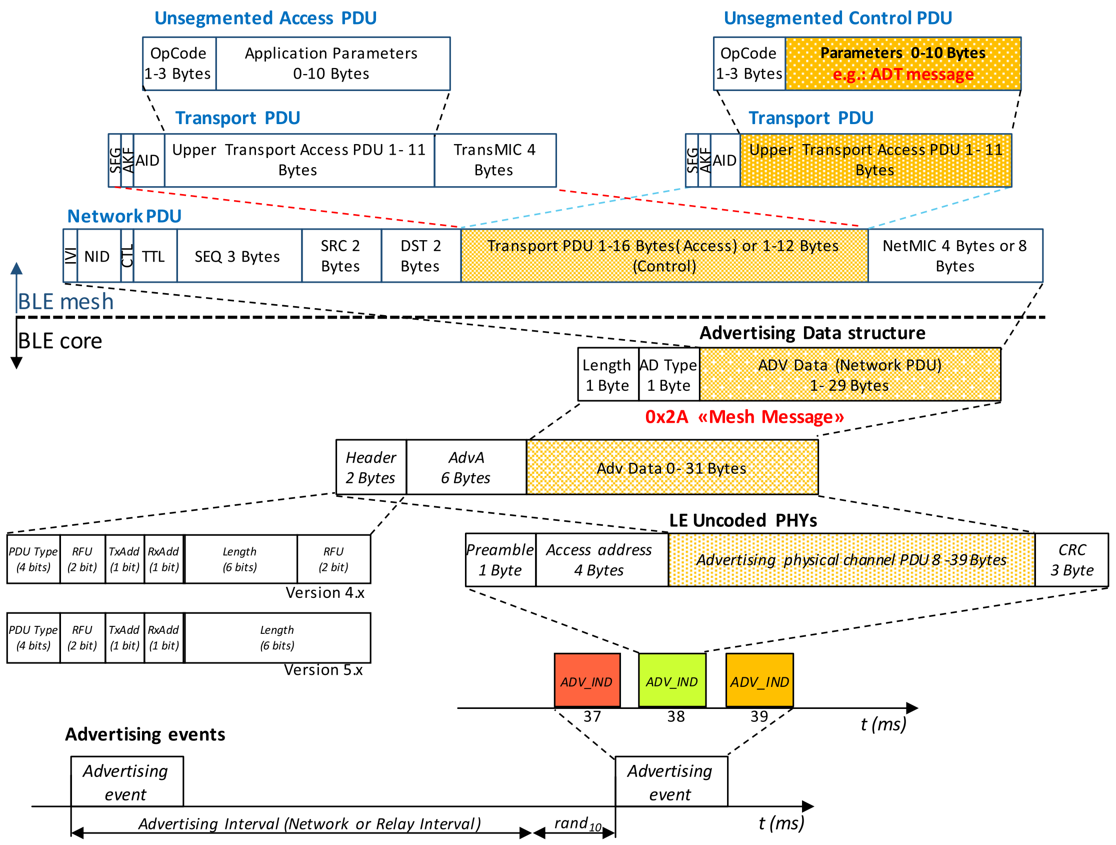


Fig. 2. BLE mesh message encapsulation and use of BLE *non-connectable and non-scannable advertising events* to send mesh messages.

network PDU is transmitted between one and eight times ($N_{TC}+1$). The *netInt* interval, being equal to (N_{TIS}+1)·10 ms, takes values from 10 to 320 ms using 10 ms steps. However, the minimum *netInt* is 20 ms due to the restriction on the advertising interval fixed by the used bearer.

Relay retransmit. It is also defined at the network layer but linked to the relay feature. It controls how many times a packet should be retransmitted by a relay and the interval between them. This is controlled by the *Relay retransmit count* (R_{RC}) parameter and the network interval (*relInt*), which depends on the *Relay retransmit interval steps* (R_{RIS}) parameter (5-bit value). R_{RC} takes values between zero and seven, so a network PDU is retransmitted between one and eight times ($R_{RC}+1$). *netInt* interval, being equal to (R_{RIS} +1)·10 ms, takes values from 10 to 320 ms using 10 ms steps. However, the minimum *netInt* is 20 ms due to the restriction on the advertising interval fixed by the used bearer.

Random delay: To avoid collisions, the mesh profile also requires introducing a random delay at the link layer, controlled by the network layer. Each transmission managed by the network transmit or relay retransmit parameters should be perturbed by a random value from 0 to 10 ms from the previous transmission.

Also, and although the advertising bearer does not include

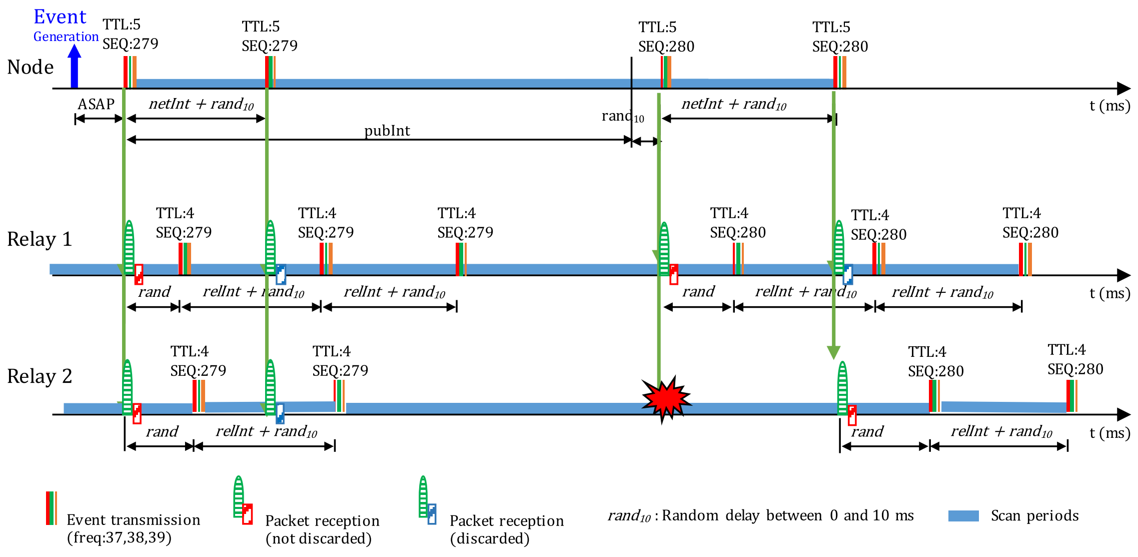


Fig. 3. Example of the publish/relay retransmit and network transmit mechanisms.

acknowledgments, BLE mesh supports transmission of unacknowledged and acknowledged messages at the model-related layer. In this case, the network schedules the transmission of the acknowledgment messages after a random delay ($rand_{ACK}$) to reduce the probability of collisions.

Fig. 3 shows an example of message transmission with BLE mesh. The source node has a P_{RC} =1 and N_{TC} =1, and the relay parameters are R_{RC} =2 and R_{RC} =1 for Relay 1 and Relay 2, respectively. The source generates a packet as soon as possible whenever a user event is generated. The first transmission is received on both relays, stored in their cache, and retransmitted R_{RC} +1 times. The second transmission is received but not retransmitted because it has the same sequence identifier as the first. The source transmits another packet with a different sequence identifier, completing the publish procedure. The Relay 1 receives it and proceeds to retransmit in the same way as the first. The packet is incorrectly received by the Relay 2. The second transmission is received on both relays and is only retransmitted by the Relay 2. Note that, later, we will suppose that these variables, can be configured differently for transmitting the data or the control message. So, hereafter, we use superscripts to indicate this difference. Additionally, from now on, every time-referred equation, variable, or parameter is expressed in ms.

Fig. 2 also illustrates the complete encapsulation process from the application layer to the bearer layer considering ADV_NONCONN_IND PDUs based advertisements. The bottom half of the figure is dedicated to explaining the advertising bearer operation. Reading the top half of the figure (mesh stack), there are two message types allowed: access and control messages. When the message to be transmitted is large, no matter its type, it can be segmented. An unsegmented access/control message can convey up to 10 bytes of data, which is enough for the purposes considered in this paper. The difference in the encapsulation of an access versus a control message comes in the transport and network layers. An access message divides the Message Integrity Checks (MICs) into two 4-byte parts, one in the transport layer and the other in the network layer. This is done to allow the encryption of data application messages. For example, a relay may retransmit a message if it belongs to the network (Network MIC OK), but not necessarily has to know its contents (Transport MIC KO). Control messages, instead, are critical messages used to manage the network. Thus, they also use eight bytes, but all of them are dedicated to protecting the network PDU. In this way, both message types have the same length at the network layer. This network PDU is introduced into a standard BLE advertising data structure with a defined advertisement type: 0×2A for mesh messages or 0×2B for mesh beacons. Here on, the advertisement is passed to the physical layer as any other BLE advertisement.

Out of current mesh specification, the new proposal includes the use of *scannable undirected advertising* events, as a part of the poll/interrogation process before mesh flooding reactivation. Thus, we proceed to describe the event. In this case, for every advertising event, the advertiser device broadcasts advertising information (ADV_SCAN_IND PDU) in sequence over each of the three advertising channels (index=37, 38, and 39). When an ADV_SCAN_IND packet is received by a device configured in *active scan* mode, the scanner device (after T_{IFS}=150 μs) is allowed to respond with a scan request (SCAN_REQ PDU) to request additional information from the advertiser. After the reception of a SCAN_REQ, the advertiser device responds T_{IFS} later with the corresponding scan response (SCAN_RSP PDU) on the same advertising channel index. Once the SCAN_RSP PDU is sent, the advertiser shall either move to the next used primary channel to send another ADV_SCAN_IND PDU or close the advertising event as illustrated in Fig. 4. Notice that ADV_SCAN_IND and SCAN_RSP may transport an access or control mesh message, whereas SCAN_REQ has only been processed at the link layer.

3. PSM-DMO proposal

The proposal is a scan and transmission scheme adapted to the BLE mesh specification, which allows all nodes of the mesh network to go into scan mode only when there is a node with data to transmit and flooding is required. The proposal avoids applying a continuous scan scheme or as close as possible to 100 % of the time, as defined in the standard.

The main idea is to replace the continuous scan with a periodic but asynchronous polling process whereby the nodes interrogate their neighbors about the existence of data to receive or to retransmit through the network. Nodes remain by default in sleep mode between polls to save energy and they only go into scan mode during the period of time the mesh network will be involved in the transmission and dissemination of data. This period is estimated by the node which is the source of data, it is announced to its neighbors and it is propagated consecutively by all the relay nodes until the destination.

Actually, the proposal is defined as an optional operation mode,

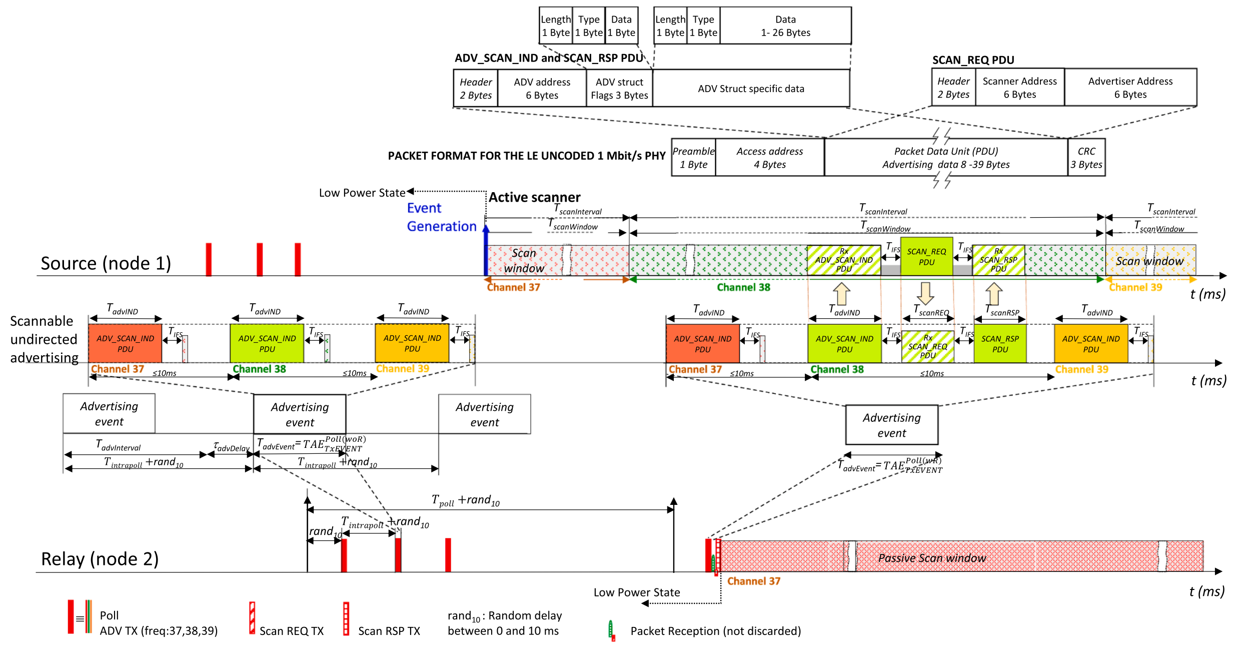


Fig. 4. Poll/interrogation process. Based on *scannable undirected advertising events*.

eligible when nodes operate in scenarios with applications that are delay tolerant and/or make infrequent transmissions. The aim is to reduce the consumption in the network relays.

The objective of the proposal differs from the concept of Bluetooth Mesh Low Power Node (BM-LPN) and the associated friendly node function, already defined in the standard. According to the specification, BM-LPN nodes are not responsible for message relaying but they can operate within a mesh network at significantly reduced reception duty cycles using a friendship relationship to another node, called "Friend Node". BM-LPN can transmit messages at any time since it is assumed that at least one of its next-hop neighbor devices will be always ready to receive and forward such messages. However, a friend node (that is, a relay node supporting the friend feature) assists the BM-LPN in reception, by storing messages destined to the BM-LPN while this node is in a sleep state. Forwarding of messages by the friend node occurs on demand when the BM-LPN asynchronously polls their friend for messages awaiting delivery. After sending the BM-LPN request to its friend node, BM-LPN returns to sleep node for a *receiveDelay* time (which allows the friend node to prepare its response) and then, it spends a *receiveWindow* time listening to the channel for a potential response.

The current proposal is also based on a polling/request mechanism but differs significantly in the concept and procedure described before. In this case, the objective is that all nodes stay in sleep mode most of the time. It limits the scan mode in all the nodes to the periods where incoming packets are required to be flooding through the network.

The proposal is compatible with the underlying core BLE since specification v5.0 [35]. The poll/request process is implemented with the support of connectable and scannable undirected advertising events, currently out of the mesh specification as potential bearer [15]. On the other hand, notifications about the time the mesh network is expected to be busy forwarding messages are performed using non-connectable and non-scannable advertising events and according to BLE mesh. We name these notifications *Awaiting Data and Transmission* (ADT) message.

The main objectives of this proposal coincide with the BLE mesh asynchronous dynamic scan mechanism (BMADS scheme) proposed by the authors in [22] to reduce the overall energy consumption of the mesh networks. However, the overall mechanism differs significantly. BMADS feature reduces the scan cycles of the nodes by defining a low energy scan duty cycle while there is nothing to transmit. That is, being *scanWin* and *scanInt* the values of the scan window and scan interval at the receiver, respectively, BMADS uses $scanWin < scanInt$, instead of keeping the receiver in continuous scan mode ($scanWin = scanInt$). To prepare the network for data transmission, BMADS sends a new control message sequence (ADS message) that puts the nodes into continuous scan mode before transmitting the data (ADS specifies the time they should stay scanning continuously). Fig. 4 in [22] shows the flow of messages associated to the proposal. The main issue is that *scanWin* and *scanInt* should be configured to reduce the energy consumption but ensuring the reception of at least one of the ADS message sequence packets sent from any neighbor. There is a tradeoff between scan cycle configuration and the required length of the ADS sequence configuration to ensure reliability. In fact, there is not a single optimal configuration, seen as the compromise between the energy saving and the additional delay introduced. BMADS provides significant energy savings, which depends on applications, size of the network, and scenarios. Up to 98% of energy savings in selected cases (applications that can tolerate delays of tens of seconds in low-hop networks, i.e. 4 hops, and sporadic transmissions, i.e. TX/day), although extreme cases of networks requiring 127 hops and having high traffic can still show savings of 25%. However, where several transmissions of an ADS message within the scan periods are required to ensure a successful control message reception in non-ideal condition due to collisions and noise, energy saving is reduced (up to 95% with three required transmissions for TX/day and 4 hops). Energy saving is significant but, there is a continuous energy consumption related to the scan periods in addition to energy consumption linked to ADS message sequence transmission. It can be thought that the consumption associated with the duty cycle of the scan process may seem negligible compared to transmissions (i.e. poll or control message transmissions like ADS) but this assessment is inexact. Note that, to optimize the consumption, it is necessary to minimize *scanWin*, but the minimum *scanWin* is 10 ms when non-scannable advertising events are used.

The new proposal (*PSM-DMO*) presented here replaces the requirement of periodic scan cycles with a periodic polling scheme. Certainly, polling based on *scannable undirected advertising events* also requires that nodes enter in a RX scan period after the ADV_SCAN_IND transmission and a T_{IFS} period. However, this scan period is equal to 60 μs if the neighbors do not respond with a SCAN_REQ, being significantly lower than the 10 ms of the BMADS proposal, as previously described. It was presented in Nordic Semiconductor [36] and measured experimentally by the authors in [37] and [38]. Additionally, it will also be visualized in Figs.10 and 11 in Section 4. The energy consumption linked to the periodic polling is effectively lower than scan cycles proposed in BMADS.

In any case, same as BMADS, a main characteristic of the proposal is that it does not require modifying the physical and MAC layer of the core BLE specification. It does not either require relevant changes in the mesh protocol stack structure although an additional feature is required. Fig. 1 depicts the modified BLE mesh stack including the new feature (PSM: Power Save Mode and DMO: Discontinuous Mesh Operation) in dark orange. The necessary changes concern the definition of a new control command linked to the ADT transmission and its management, the activation, control and deactivation of commands linked to the polling process and handling of timings linked to mesh activation and transmission. Note that *scannable undirected advertising events* are not currently used as a bearer for access or control PDUs. The new commands may be identified using the RFU fields available in the Header field in the current BLE standard frame definitions, showed in Fig. 2. It is necessary that all nodes in the mesh network should include this feature for PSM-DMO to operate. If there are nodes that do not support polling and ADT transmission, network should work in standard mode.

Next, we describe the general phases of the mechanism. Then, the overall algorithm will be summarized in Fig. 7.

3.1. General phases of the mechanism

The general phases of the PSM-DMO process can be concisely summarized as follows. Nodes are usually in low power state. In this state, all nodes sleep and send regular poll messages to see whether there is traffic to be relayed (box in grey color in Fig. 7). When a node has traffic to send or retransmit, it listens (box in red color in Fig. 7) for such poll messages coming from neighboring nodes and makes use of them to activate these neighboring nodes and put them in scanning mode. This process continues until the entire network is awake. The process includes the transmission of information (box in green color in Fig. 7) about the time required by the nodes to be in active state. Once the network is awake, data transmission occurs (box in blue color in Fig. 7) in a similar way than in the standard mesh.

Next, we describe step by step the basis of the implementation. General parameters included in the proposal are summarized in Table 1.

1) *Low Power State, Interrogation Process*

By default, all the nodes remain in a sleep mode. That is, scanning is interrupted, but nodes transmit periodic and asynchronous poll requests to neighbor nodes to determine if they have to change to scan mode. This mode of operation is only active when nodes do not have data to transmit, nor to retransmit nor they are also expecting to receive them from other nodes (in case they are relays).

The interrogation process is based on the periodical activation by each node (advertiser) of *scannable undirected advertising events*. The visual representation of the poll process is shown in Fig. 4, with details about required timing relationships between transmissions. The polling

Table 1
Parameters used in the PSM-DMO proposal.

Parameters	Description
Generic random delays	
rand	Undefined random value in the specifications.
$rand_{10}$	Random value, uniformly distributed between 0 and 10 ms.
$rand_{MAX}$	20 ms. Upper limit in some random variables.
Scanning parameters	
scanInt	Generic scan interval.
scanWin	Generic scan window.
Generic processing times	
T_{PROC}^{relay}, T_{PROC}^{sink}	Processing times of the relays and sink, respectively.
Poll process	
Poll event	Poll event defined as a scannable undirected advertising event.
$T_{ADVpoll}$, $T_{scanREQpoll}$ $T_{scanRSPpoll}$	Times linked to advertising, request and response packets in the scannable undirected advertising event.
T_{poll}	Fixed time between consecutive polling processes (composed by only one polling event). The time between consecutive polling event is $T_{poll}+ rand_{10}$.
N_{PO}	Number of (T_{poll} + *10* ms*)* intervals to computer T_{listen}.
T_{listen}	Discovery interval: minimum interval to discover neighboring nodes. It is equal to $N_{PO}\cdot(T_{poll}+ 10)$.
ADT message transmission parameters	
ADT message	Message that includes T_{ADT}.
adv^{seqADT}	Non-connectable and non-scannable undirected advertising events to notify T_{ADT}.
$adv^{seqADT,i}$	adv^{seqADT} linked to source *i*.
$rand_{seqADT}$	Random time applied by the relay nodes after T_{listen} and before transmitting the first ADT message of the adv^{seqADT} sequence.
max($rand_{seqADT}$)	Upper limit of $rand_{seqADT}$. It is equal to rand_{MAX}.
$netInt^{seqADT}$	Network transmit interval linked to adv^{seqADT} advertisement event.
P_{RC}^{seqADT}	Public retransmit count linked to adv^{seqADT}. Set to 0.
N_{TC}^{seqADT}	Network transmit count linked to adv^{seqADT} advertisement event.
TTLseq	TTL of adv^{seqADT} advertising event. Set to 0.
$TAE_{TxEVENT}^{seqADT}$	Duration of the full adv^{seqADT} advertising event.
T_{seqADT}	Maximum time needed for the transmission of the sequence of ADT messages.
T_{ADT}	Awaiting Data and Transmission time: amount of time a node should remain in scan state awaiting data. It is included as a parameter in the *ADT* message.
T_{ADT1}	Estimation of the maximum time a node must wait into scan mode before starting to receive data.
T_{ADT1}^{source}	Estimation of T_{ADT1} computed by the source.
T_{ADT2}	Estimation of maximum time needed to transmit the data
TTL_{ADT}	TTL parameter included as a parameter in the *ADT* message.
PId	Process Identifier ($PId_{,j}$ is the *PId* linked to the source j).
$TIMEOUTRX_{seqADT}$	Timeout to detect reception of adv^{seqADT} (applied in relay/sink). After that, the polling process is reactivated.
Data transmission parameters	
$pubInt^{data}$	Publish retransmit interval
$netInt^{data}$	Network transmit interval
$relInt^{data}$	Relay retransmit interval
P_{RC}^{data}	Publish retransmit count
N_{TC}^{data}	Network transmit count
R_{RC}^{data}	Relay retransmit count
$TAE_{TxEVENT}^{data}$	Duration of the advertising event linked to data transmission.

event consists of sending packets named ADV_SCAN_IND through one of the three advertising channels defined in the specifications (in this case on all three, CH=37, CH=38, and CH=39). Note that, in Fig. 4, both nodes 1 and 2 are in low power state in the first period of the time. Nodes leave the low power state if a data transmission event is generated or activated by them (e.g. node 1 in Fig. 4), initiating the process that propagates the switch of all the nodes to a continuous scan. Neighboring nodes in polling mode could start receiving responses from active scan devices. Note that difference between passive and active scan is that an active scanner can request additional information from the advertiser while the passive scanner only listens. That is, any device in active scan (e.g. initially, the source node in Fig. 4 after event generation), receiving an ADV_SCAN_IND from an advertiser (polling node), responds with a unicast scan request PDU (SCAN_REQ) to the advertiser on the same channel where it has been received and then listens for the scan response PDU (SCAN_RSP) on the same advertising index. The scanner only sends the SCAN_REQ if the advertiser address is allowed by a scanner filter policy defined in it. In the same way, the advertiser sends the SCAN_RSP only if the SCAN_REQ is received from a scanner allowed by the advertising filter policy defined in the advertiser. Scan and advertiser filters determine, respectively, the devices for which advertising packets will be processed by the scanner and devices for which the advertiser will accept a scan request. Once the SCAN_RSP PDU is sent or if the advertising filter policy forbids processing the SCAN_REQ, the advertiser shall either move to the next used primary channel index to send another ADV_SCAN_IND or close the advertising event. In this proposal, no restrictions are applied concerning the possibility of discovering several devices that send data into two or the three different frequencies within the same event. Therefore, the proposal contemplates closing the event after finishing the round of the three frequencies. However, this scenario is extremely unlikely and, if possible, forces the length of the scan period to be managed to consider the coexistence of multiple parallel transmissions. This aspect will be reviewed later.

After sending the sequence of ADV_SCAN_IND linked to the polling event, if the requesting node (relay or sink, node 2 in Fig. 4) has received a SCAN_REQ on any of the three frequencies, it sends the SCAN_RSP PDU. Then, the requesting node goes temporarily into continuous passive scan state as close as possible to 100% ($T_{scanWin}$=$T_{scanInt}$) and deactivates its polling process. Reasons for switching to passive scan instead of active scan are discussed later.

After entering in passive scan state, the link layer should listen on a different primary advertising channel index (there is not a selection rule for the first index) for the duration of each consecutive $T_{scanWin}$. For example, in Fig. 4, node 1 responds to node 2 in CH=38, however, node 2 starts the passive scan in CH=37 according to its own programmed sequence for each $T_{scanInt}$.

The time between consecutive polling processes is T_{poll} plus a random period, which we have defined between 0 and 10 ms ($rand_{10}$). The polling process is composed of only one polling event. This allows to configure the process as a periodic advertising event, being the time between the start of two consecutive advertising events ($T_{advEvent}$) computed as $T_{advInterval}$=T_{poll} plus $\tau_{advDelay}$=$rand_{10}$.

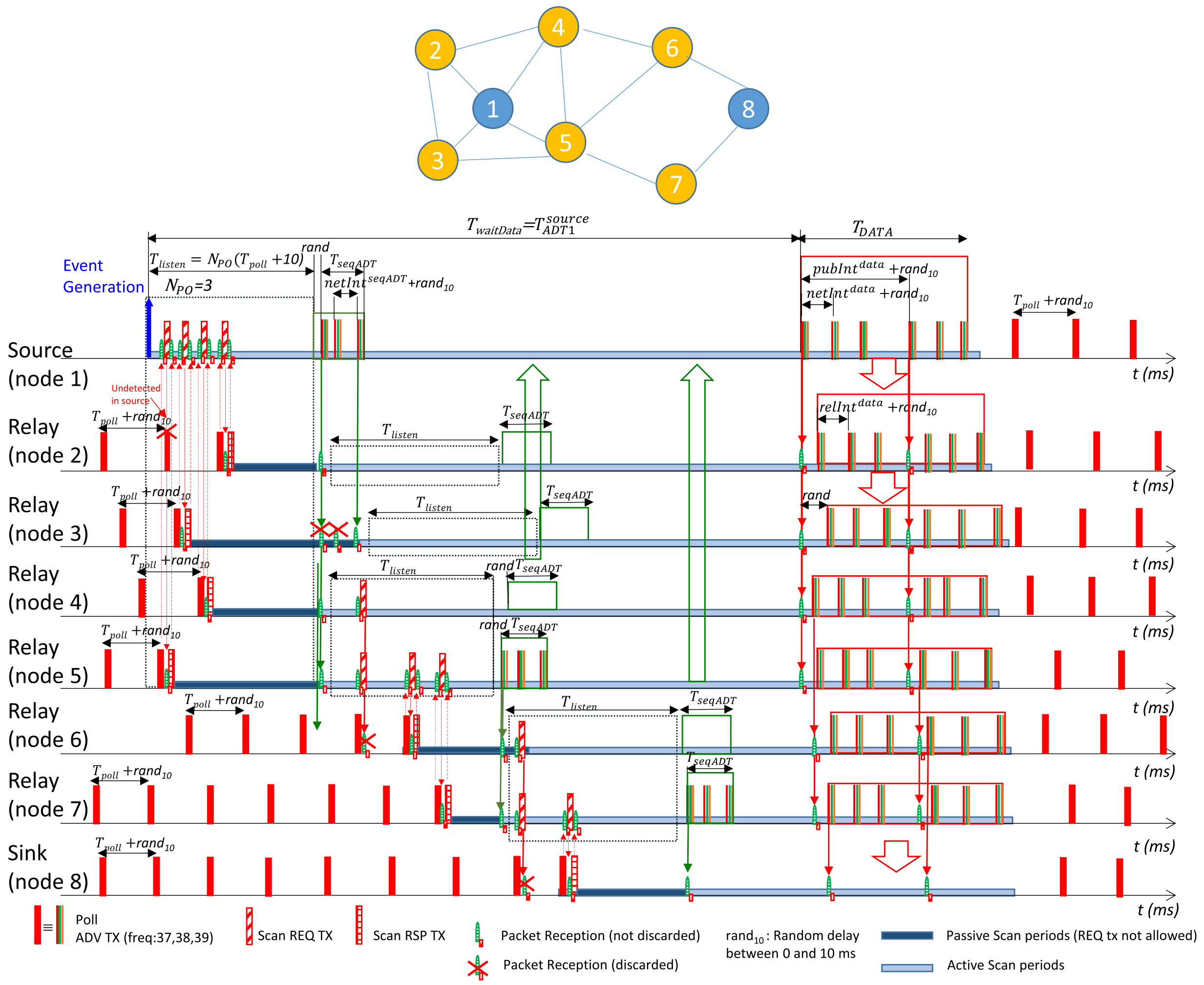
The process described above is not aware of the source or the destination of the messages that will be sent later. It is only a simple mechanism that allows nodes to switch to the required scan mode (firstly a passive mode and then an active scan mode).

2) *Activation and management of the scan periods. Cascade spread of the process*

Once we have defined the more general basis concerning the polling mechanism, let's specify how the activation of the scan periods and transmissions are managed, first in the source node and then in the relays and destination nodes. Fig. 5 shows an example of the whole mechanism to support the description.

2A) *Mechanism at the source node*

A source node, with data to transmit, switches to scan mode awaiting to receive an ADV_SCAN_IND from each of its relay neighbors according to the procedure described above. When it receives an ADV_SCAN_IND from a valid address and, then, the corresponding SCAN_RSP after sending the SCAN_REQ, the source knows the relay/sink node that interrogated has moved to passive scan mode. The same does not happen in case there is no SCAN_RSP. In this case, the source only knows that a



Event Generation
Source (node 1)
Relay (node 2)
Relay (node 3)
Relay (node 4)
Relay (node 5)
Relay (node 6)
Relay (node 7)
Sink (node 8)
$T_{waitData}=T^{source}_{ADT1}$
T_{DATA}
$T_{listen}=N_{PO}(T_{poll}+10)$
N_{PO}=3
T_{seqADT}
$netInt^{seqADT}+rand_{10}$
$pubInt^{data}+rand_{10}$
$netInt^{data}+rand_{10}$
$relInt^{data}+rand_{10}$
$T_{poll}+rand_{10}$
Undetected in source
t (ms)
Poll ADV TX (freq:37,38,39)
Scan REQ TX
Scan RSP TX
Packet Reception (not discarded)
Packet Reception (discarded)
$rand_{10}$: Random delay between 0 and 10 ms
Passive Scan periods (REQ tx not allowed)
Active Scan periods

Fig. 5. Example of the proposed mechanism.

neighbor exists. How to manage this issue will be discussed later.

In absence of errors in reception due to channel effects or collisions, after a time equal to T_{poll} plus 10 ms (maximum of $rand_{10}$), the source should have been successfully interrogated by all its neighbor relays (if we consider that all relays are configured identically) and knows they all moved to passive scan.

To combat the effects of channel errors and collisions, instead of waiting for a minimum time T_{poll}+10 ms, the minimum discovery window for neighboring nodes can be extended to an N_{PO} number of T_{poll} + 10 ms intervals. We name this period T_{listen}. For instance, if N_{PO}= 2, T_{listen}=2·(T_{poll} + 10). The N_{PO} value should be estimated based on the expected collision probability for the application scenario and configuration parameters of poll process, in addition to the Block Error Rate (BLER). N_{PO} will then be configured the same in all nodes of the network.

Once the T_{listen} period elapses, the source cannot start sending data yet. First, it is required that the process is repeated in the relays to propagate it over the network so that all relay nodes and the sink in the mesh network have changed and remain in active state. The objective is to prepare the network for the transmission of data. For this purpose, it is required that each relay node, entering into continuous scan mode, receives a mesh control field/message which should specify the amount of time it should remain in scan state awaiting data. This time interval, we denoted as T_{ADT} (Awaiting Data and Transmission time), is composed by two components (T_{ADT1} and T_{ADT2}) we describe later and is calculated at the source node considering the worst-case scenario of delay.

It is reasonable to think that the simplest and most desirable approach to send the T_{ADT} value could be encapsulated inside the SCAN_REQ, but this option is not feasible, since this packet does not include any data field (see in Fig. 4 the SCAN_REQ PDU) and it has no bits available in the header. So, the proposal is to send mesh messages to notify the T_{ADT} (named *ADT message*) using *non-connectable and non-scannable undirected advertising events* (named *adv*^{seqADT}), once the T_{poll} + 10 ms period has elapsed (see in Fig. 5 the source). The network layer address field should be set to the broadcast fixed group address 0xFFFF to allow the processing of the message by every node that receives it. *ADT* message does not include any information about the specific destination of the data flow expected to be transmitted. Nevertheless, *ADT* message is sent using the mesh bearer layer. It is a control message, defined and managed at the new feature (PSM-DMO) introduced in the mesh protocol stack, which goes through the mesh protocol stack as illustrated in Fig. 2. Details about *ADT* parameters will be introduced next.

To combat the effects of channel errors and collisions over *ADT* messages, redundancy by repetition is applied using, in this case, mechanisms defined in the mesh specification. Each *ADT* message should be processed only by neighbors of the node transmitting the message. *ADT* message is never relayed because this packet should not be retransmitted immediately on the receiving nodes. Thus, the TTL field (identified as TTL_{seq}) is fixed to zero. When a neighbor propagates *ADT* information to its neighbors in the next hop, the *ADT* message should also be regenerated. Specifically, the repetition of the *ADT* message follows the standard network transmit procedure, with a N_{TC} and *netInt,* denoted as N^{seqADT}_{TC} and *netInt*^{seqADT}, respectively. In this way, each ADV packet, containing the *ADT* information, is transmitted N_{TC}+1 times in each of the three frequency channels (37, 38, and 39) using the ADV event. On the other hand, the maximum time needed for the transmission of the sequence of *ADT* messages is given by the following equation:

$$T_{seqADT} = N_{TC}^{seqADT} \cdot \left(netInt^{seqADT} + 10 \right) + TAE_{TxEVENT}^{seqADT} \tag{1}$$

where $netInt^{seqADT}$ plus 10 ms (the maximum of a random interval between 0 and 10 ms -denoted as $rand_{10}$-) is the maximum time interval between consecutive adv^{seqADT} events and $TAE_{TxEVENT}^{seqADT}$ the duration of the full advertising event (transmission of the ADT message on all three frequencies).

After elapsing the T_{listen} and transmitting the sequence of ADT messages, if neighbor nodes are detected, the source node cannot transmit data immediately. It should wait enough time, named as T_{ADT1}, until all nodes in the network are in scan mode before start sending the data. This time must be added to the maximum time, named as T_{ADT2}, required to transmit the data through the network. Data acknowledgment from the destination node is not considered in this work, but it can be easily contemplated with few modifications. In this case, the source will use larger values for ADT2, including the estimation of ACK transmission.

Both values, T_{ADT1} and T_{ADT2}, must be reported in the *ADT* message. As we said above, the *ADT* message does not include any information about the specific destination of the data stream expected to be transmitted.

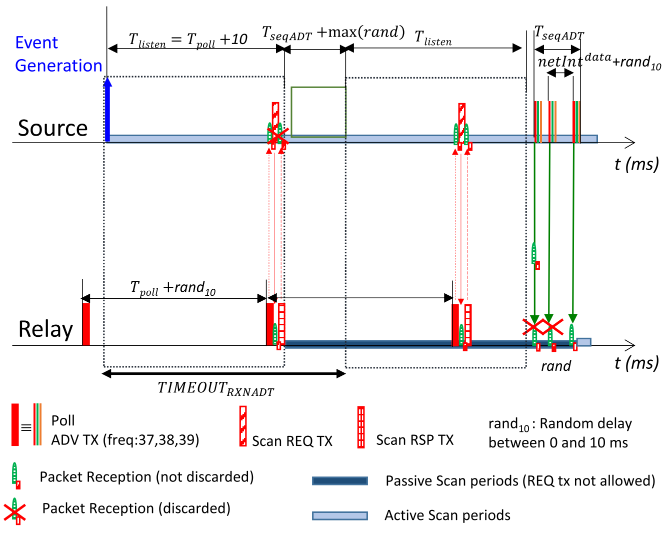
2B) *Mechanism in the relays and the sink nodes*

In the relay and sink nodes the process is repeated similarly but the following peculiarities need to be considered.

In relation to the polling state, as mentioned above, after sending an ADV_SCAN_IND, if a relay node receives a valid SCAN_REQ PDU (there is a neighbor that requires transmission of data), it sends the SCAN_RSP PDU, disables the polling process, and proceeds to continuously scan on the frequency corresponding according to the configuration of the scan procedure. However, when the node enters the scan mode, it must still inhibit the response to polling messages from other nodes for a period (see, e.g., dark blue periods in node 2 in Fig. 5). Specifically, the inhibition lasts until it does not successfully receive an *ADT* message of the sequence of adv^{seqADT} events from the node that put it in scanning mode. Thus, passive scan is used instead of active scan. There are two reasons for that:

(1) Receiving an adv^{seqADT} is the only explicit evidence that the interrogated node has successfully received the SCAN_RSP PDU from this relay/sink node or any of its neighbors during the T_{listen} period. Let's consider a relay/sink node that is the only neighbor of the one sending SCAN_REQ. If the interrogated node did not receive the response (SCAN_RSP), it is not sure that the relay/sink node has entered scan mode, since it is not possible to know if the SCAN_REQ PDU was received correctly. Therefore, the interrogated node will not send the sequence of adv^{seqADT}.
(2) If a relay node immediately goes into active scan mode, it can respond to the polling request from other neighboring nodes that are also one hop away from the node from which they expect to receive the data. This could be the case that occurred in node 5 with respect to nodes 3 and 4 in Fig. 5. Therefore, collisions between *scannable undirected advertising events* are unnecessarily increased.

To solve the anomalies derived from the loss of the SCAN_RSP packet, we define a timeout, called $TIMEOUTRX_{seqADT}$, which is particularly useful when the node is the only neighbor of the interrogated node. If after sending SCAN_RSP a relay/sink node does not receive any adv^{seqADT} within a $TIMEOUTRX_{seqADT}$ period equal to $T_{listen}+T_{seqADT}+max(rand)$, the node reactivates the polling process. In this case,



Event Generation
$T_{listen} = T_{poll} +10$
T_{seqADT} +max(rand)
T_{listen}
T_{seqADT}
$netInt^{data}$+$rand_{10}$
Source
t (ms)
T_{poll} +$rand_{10}$
Relay
rand
t (ms)
$TIMEOUT_{RXNADT}$
Poll
ADV TX (freq:37,38,39)
Scan REQ TX
Scan RSP TX
$rand_{10}$: Random delay between 0 and 10 ms
Packet Reception (not discarded)
Packet Reception (discarded)
Passive Scan periods (REQ tx not allowed)
Active Scan periods

Fig. 6. $TIMEOUTRX_{seqADT}$ to solve non-detection.

polling and continuous passive scan coexist but responses (thanks to passive scan mode) to its own polling neighbors are inhibited. This solution allows that in case this node was the only neighbor of a given node, the propagation of the process of switching to the scan mode is not blocked in the network. Really, this situation is expected to occur with low probability, especially when the number of poll events in the T_{listen} period is greater than unity and this number is correctly dimensioned. The scan process can be disabled if nothing is received after a second T_{listen} period and the node returns to the low power state. Fig. 6 shows an example of the situation. Further details about the use of this $TIMEOUTRX_{seqADT}$ and the concatenation of various T_{listen} periods will be discussed later, related to anomalies management.

Once a relay node successfully receives the adv^{seqADT}, it starts a new time period in active scan mode equal to the ADT value received in adv^{seqADT}, being now able to respond to polling neighbors to facilitate the pass to the scan mode of the rest of the network nodes (see, e.g., light blue periods in node 2 in Fig. 5. After that, similarly as the source did, at the beginning of that scan period and for a time T_{listen}, the relay waits for the reception of ADV_SCAN_IND PDUs. When an ADV_SCAN_IND is received, the relay responds with a SCAN_REQ and waits for the SCAN_RSP. After T_{listen}, if at least one SCAN_RSP has been received, the relay sends the adv^{seqADT} with the same procedure, contents, and parameterization used by the source. Then, it remains in active scan, waiting for the reception of the data in mesh mode.

A small random delay should be introduced between finishing the T_{listen} and relaying the ADT message using the adv^{seqADT} to reduce collisions of the first adv^{seqADT} events when multiple relays have been involved in the T_{listen} period at the same time (e.g., see nodes 4 and 5 in Fig. 5). This random time (named $rand_{seqADT}$ but represented as rand in figures) is similar to the one defined in the mesh for the ACK in relay nodes. Specification [15] says that $rand_{ACK}$ should be between 20 and 50 ms when the ACK is the response to a message that was sent to a unicast address, but between 20 and 500 ms to reduce the probability of collisions, if it was sent to a group address or a virtual address. In this case, $rand_{seqADT}$ must be set according to the expected probability of collisions.

Once the relay node has sent the sequence of adv^{seqADT} events, it expects to remain in active scan mode for a period of time equal to that specified in the adv^{seqADT} that it had received from its neighboring nodes.

It would be feasible to define a second $TIMEOUTRX_{data}$ to disable the scan process if nothing is received after a T_{ADT} period.

Above, we have described the basis of the process by which all network nodes enter into scan mode, to proceed below with data transmission in mesh mode. In addition to Figs. 5 and 6, the algorithm of

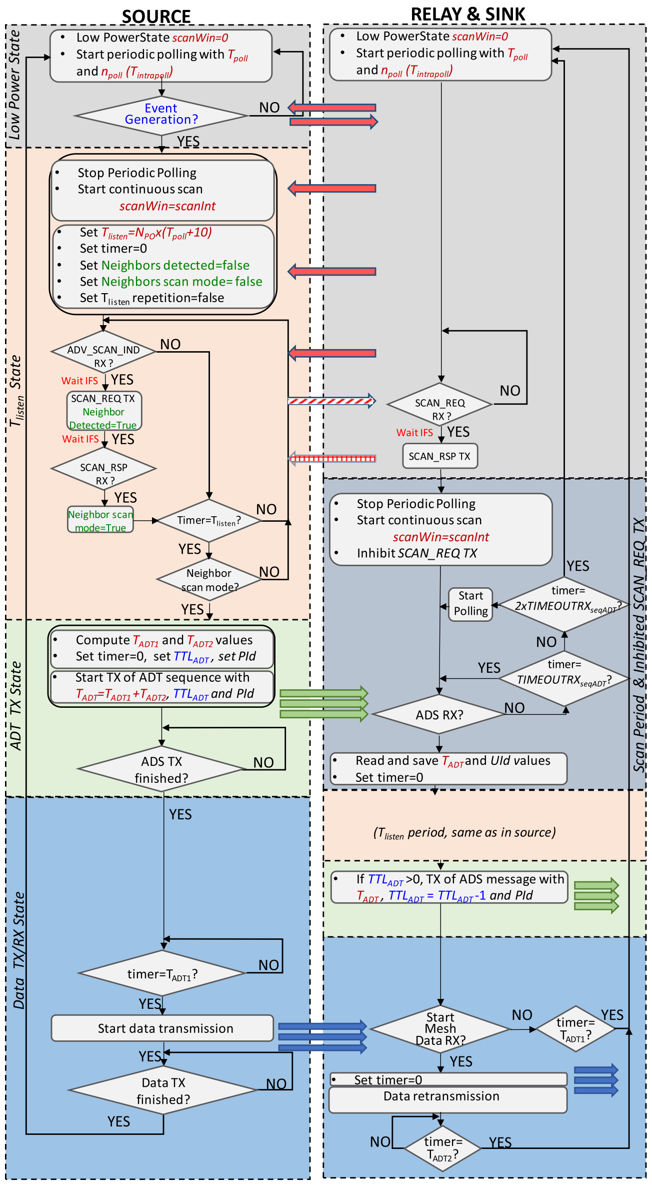

SOURCE
RELAY & SINK
Low Power State
Low PowerState scanWin=0
Start periodic polling with T_{poll} and n_{poll} ($T_{intrapoll}$)
Event Generation?
NO
YES
T_{listen} State
Stop Periodic Polling
Start continuous scan scanWin=scanInt
Set T_{listen}=N_{PO}x(T_{poll}+10)
Set timer=0
Set Neighbors detected=false
Set Neighbors scan mode= false
Set T_{listen} repetition=false
ADV_SCAN_IND RX ?
Wait IFS
SCAN_REQ TX Neighbor Detected=True
SCAN_RSP RX ?
Neighbor scan mode=True
Timer=T_{listen}?
Neighbor scan mode?
SCAN_REQ RX ?
SCAN_RSP TX
Inhibit SCAN_REQ TX
Start Polling
timer= 2xTIMEOUTRX$_{seqADT}$?
timer= TIMEOUTRX$_{seqADT}$?
ADS RX?
Scan Period & Inhibited SCAN_REQ TX
ADT TX State
Compute T_{ADT1} and T_{ADT2} values
Set timer=0, set TTL_{ADT}, set PId
Start TX of ADT sequence with T_{ADT}=T_{ADT1}+T_{ADT2}, TTL_{ADT} and PId
ADS TX finished?
Read and save T_{ADT} and UId values
Set timer=0
(T_{listen} period, same as in source)
If TTL_{ADT}>0, TX of ADS message with T_{ADT}, TTL_{ADT} = TTL_{ADT}-1 and PId
Data TX/RX State
timer=T_{ADT1}?
Start data transmission
Data TX finished?
Start Mesh Data RX?
timer= T_{ADT1}?
Set timer=0
Data retransmission
timer= T_{ADT2}?

Fig. 7. PSM-DMO general algorithm.

Fig. 7 helps to illustrate the mechanism. However, additional details must be considered, completing the procedure that will be now described in Section 3.2.

3.2. Additional issues of the mechanism

A) ADT message parameters and calculation of T_{ADT}

As we refer to above, to fulfill the approach, each *ADT* message should be processed only by neighbors of the node transmitting the message. *ADT* message is never relayed immediately. Instead, it is regenerated each hop. Thus, the standard *TTL* (*TTL$_{seq}$*) field present in the mesh message is fixed to zero, being the network layer address field the broadcast fixed group address 0xFFFF.

The main parameters of the ADT message concerns to T_{ADT} (being the pair T_{ADT1} and T_{ADT2}), TTL_{ADT} and *PId,* all of them described subsequently. T_{ADT} is the amount of time a node should be in continuous scan to guarantee the reception of the data after reception of the ADT message. T_{ADT} is calculated at the source node and it is composed by two separate components: T_{ADT1} and T_{ADT2}. T_{ADT1} is an estimation of the maximum time a node must wait into scan mode before starting to receive data. T_{ADT2} is the estimation of maximum time needed to transmit the data.

$$T_{ADT} = T_{ADT1} + T_{ADT2} \tag{2}$$

T_{ADT1} and T_{ADT2} are upper limits. As Fig. 4 shows, it is feasible for relay nodes (see nodes 6, 7 and 8) to receive the first data packet in a clearly shorter interval than T_{ADT1}. Similarly, the time a node must spend receiving data will generally be less than T_{ADT2}. Thus, both values should be sent in a disaggregated way in the ADT message to reduce the time that the nodes remain in scan mode. Therefore, once the reception of the data has started, only the T_{ADT2} applies. The algorithm of Fig. 7 illustrates this fact.

T_{ADT1} and T_{ADT2} depend on the maximum number of hops required to reach the destination. It is necessary to estimate it and to limit the range of the propagation mechanism by using a new TTL parameter (in this case, renamed TTL_{ADT}). TTL_{ADT} value can be determined previously, for instance, using a heartbeat message. The TTL_{ADT} value should be included as a parameter in the data field of the *ADT* message. Recall that TTL_{ADT} differs from the $TTL=0$ ($TTL_{seq}=0$) set at the Network PDU (see Fig. 2). When a relay must generate a new ADT message, it reduces the TTL_{ADT} by one. Nevertheless, to reduce node processing, all nodes retransmit these same T_{ADT1} and T_{ADT2} values. Note that this is done without considering that nodes that are several hops away from the source need to wait a shorter time.

From the point of view of the source node, once the packet generation event occurs, the time it needs to wait before starting data transmission is T^{source}_{ADT1}, as computed in Eq. (3):

$$\begin{aligned} T^{source}_{ADT1} &= T_{listen} + T_{seqADT} + T_{ADT1} = T_{listen} + T_{seqADT} + \\ &+ (TTL_{ADT} - 1)\cdot\left[T_{listen} + rand_{MAX} + T_{seqADT}\right] + \\ &+ (TTL_{ADT} - 1)\cdot T^{relay}_{PROC} + T^{sink}_{PROC} \end{aligned} \tag{3}$$

where T_{listen} depends on the parameters N_{PO} and T_{poll} according with Eq. (4):

$$T_{listen} = N_{PO}\cdot\left(T_{poll} + 10\right) \tag{4}$$

T_{seqADT} is the maximum time needed for the transmission of the sequence of adv^{seqADT} as computed in Eq. (1), T^{relay}_{PROC} and T^{sink}_{PROC} are the processing times of the relays and sink respectively, while $rand_{MAX} = \max(rand_{seqADT})$ is the upper limit of the random time applied by the relay nodes before the first ADT message of the adv^{seqADT} sequence.

However, the T$_{ADT1}$ value propagated to neighbors and by relay nodes is computed with Eq. (5):

$$\begin{aligned} T_{ADT1} &= (TTL_{ADT} - 1)\cdot\left[T_{listen} + rand_{MAX} + T_{seqADT}\right] + \\ &+ (TTL_{ADT} - 1)\cdot T^{relay}_{PROC} + T^{sink}_{PROC} \end{aligned} \tag{5}$$

On the other hand, the transmission of data is carried out in mesh mode, assuming a network of TTL_{ADT} hops. Reliability is obtained by the standard methods defined in Bluetooth Mesh itself. That is, repetitions which are controlled in the network and model layers through *Network transmit* (pairN^{data}_{TC} and *netInt^{data}*), *Relay retransmit* (pair R^{data}_{RC} and *relInt^{data}*) and *Public retransmit* (pair$P^{data}_{RC}$and*pubInt^{data}*) processes parameters defined in Section 2. Thus, if we want to transmit N_{data} packets, the time needed to transmit data through the network (T_{ADT2}) is computed with Eq. (6).

$$\begin{aligned} T_{ADT2} &= T_{DATA} + (TTL_{ADT} - 1)\cdot T_{REL} + \\ &+ (TTL_{ADT} - 1)\cdot T^{relay}_{PROC} + T^{sink}_{PROC} \end{aligned} \tag{6}$$

with

$$\begin{aligned} T_{DATA} &= N_{data}\cdot\left(P^{data}_{RC}\cdot pubInt^{data} + N^{data}_{TC}\cdot\left(netInt^{data} + 10\right)\right) + \\ &+ (N_{data} - 1)\cdot\left(netInt^{data} + 10\right) + TAE^{data}_{TxEVENT} \end{aligned} \tag{7}$$

and

$$T_{REL} = rand_{MAX} + R^{data}_{RC} \cdot \left(relInt^{data} + 10 \right) + TAE^{data}_{TxEVENT} \tag{8}$$

where $TAE^{data}_{TxEVENT}$ is the duration of the advertising event (time elapsed from the start of the ADV PDU transmission on CH = 37 channel to the end of the ADV PDU transmission on CH = 39 channel). The specifications indicate an undefined random delay for the first relay transmission, that here is consider equal to the selected $rand_{seqADT}$. Thus $rand_{MAX}$ is considered in Eq. (8).

The last parameter of an *ADT* message must be a Process Identifier (*PId*) determined by the source node. It corresponds to a portion of the random address that the source chooses as a source address in the SCAN_REQ after event activation. It is introduced by the source into the parameters of the first ADT message, and the relays should use the same identifier as the received message. A node that receives an ADT message with a previously received identifier should discard the message because it has already been relayed.

In summary, we propose to use three bytes for each one of the first group of parameters (T_{ADT1} and T_{ADT2}). The time resolution selected for these values is the step of CLK1 defined in Vol.2 - Section 1.1 of [16]. Thus, assuming steps of 625 μs, we can keep the network on active scan, up to 10,485.76 s. For the second parameter, TTL_{ADT}, just one byte is needed. And, for the *PId* we use the remaining three bytes to reach the maximum of an unsegmented control PDU (see Fig. 2).

B). *Identification of processes and management of parallel data transmissions*

The transition between idle to scan mode may be triggered by data generation events at different nodes (with the same or different destination nodes) which could occasionally overlap in time. Keep in mind that these situations will be extremely rare because the proposal makes sense for application scenarios where nodes (for example, sensors) transmit data very sporadically and the probability of parallel transmissions is negligible. However, if they occur, the following guidelines are considered to deal with them.

In order to distinguish the different processes, instead of using the scanner address in the SCAN_REQ PDUs (six bytes, see details in Fig. 4), the source selects a random address. This option is fully compliant with the specifications. The selected address is learned by each intermediate node and is used throughout the entire process that allows the nodes to go into scan mode. That is, the address/identifier is copied as the source address into the SCAN_REQ packets that each relay node transmits. It must be included in the data field of the SCAN_RSP that responds to the SCAN_REQ. In addition, the three least significant bytes are used as *PId* in the adv^{seqADT}. This approach is used to solve some issues:

First, if a node receives a SCAN_REQ with the address or an adv^{seqADT} with the *PId*, it previously sent, it does not process these packets, avoiding triggering redundant transmissions associated with them.

The second issue concerns the management of parallel transmissions. When a node is already in scan mode (passive or active), it will postpone any new data generation event.

If the node has not exceeded the T_{listen}, continues in the process of activating the scan mode on the rest of the nodes, using in the SCAN_REQ PDU and the subsequent adv^{seqADT} sequence the random address that it has received from its neighbor node and forwarding the received T_{ADT}. The postponed event is triggered after finishing the data retransmission of the previous one. This option is inefficient from the delay viewpoint, but it is simple and limits collisions between parallel data transmission events. Nevertheless, it does not manage or solve all situations where parallel transmissions may occur.

Let's see two separate nodes in the network (they are in a low power state) that generate a data transmission event. In parallel, they switch to the scan mode to be interrogated by their neighbor nodes about the existence of data. The process needs to be repeated and propagated

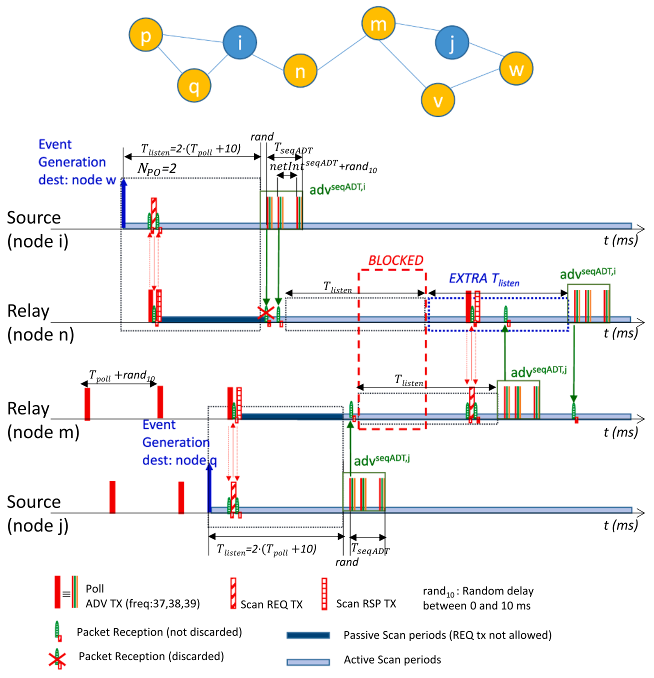


Fig. 8. Example of management of parallel data transmissions.

along with the neighbor nodes until all of them change to active scan mode. However, in this case, some anomalies appear. The process may be interrupted in the intermediate nodes located between both source nodes. The problem occurs when two neighbor intermediate nodes go into the listen period, each of them linked to one of the independent data events. Since none of them can poll the other one, these nodes are not aware of the presence of their neighboring nodes and do not proceed to send the associated adv^{seqADT} sequence. The problem is illustrated in Fig. 8, where nodes *i* and *j* are source nodes with destinations node *p* and *w*, respectively.

In this case, nodes *n* and *m* enter almost simultaneously in T_{listen} period. To deal with this anomaly, if a node has not received any ADV_SCAN_IND after a T_{listen} period, a new T_{listen} period is planned. In this new T_{listen} period, the node simultaneously scans and activates the polling process. That is, with this modification nodes (*n* and *m*) could proactively know the existence of neighbor nodes. A particular case occurs in Fig. 8. After finishing the T_{listen} period, node *m* has been polled by node *n* and it retransmits its respective $adv^{seqADT,}$ linked to source *j*, named $adv^{seqADT,j}$ with $PId_{,j}$. On the contrary, node *n* has not yet received an ADV_SCAN_IND after the new T_{listen} period because no extra T_{listen} period will be activated in node *m*. However, node *n* knows the existence of a neighbor thanks to $adv^{seqADT,j}$ reception. Thus, after finishing the T_{listen} period, node *n* sends its respective $adv^{seqADT,}$ linked to source *i*, named $adv^{seqADT,i}$ with $PId_{,i}$.

Recall that nodes need caching the parameters linked to $PId_{,i}$, and $PId_{,j}$. After receiving the second adv^{seqADT}, they will apply the new T_{ADT} threshold in the algorithm described in Fig. 7. They will be computed using the T_{ADT1} and T_{ADT2} values received from the first adv^{seqADT} (e.g. $adv^{seqADT,i}$ in node *n*, in Fig. 8) and those received in the second adv^{seqADT} (e.g. $adv^{seqADT,j}$ in node n, in Fig. 8). Specifically, T_{ADT1} and T_{ADT2} are updated conservatively according to rules specified in Eqs. (9) and (10), corresponding the superscripts (0) and (1) with the stored and second ADT message reception.

$$T_{ADT1} = \max\left(T^{(0)}_{ADT1}, T^{(1)}_{ADT1} \right) \tag{9}$$

$$T_{ADT2} = T^{(0)}_{ADT2} + T^{(1)}_{ADT2} \tag{10}$$

Note that T_{ADT1} and T_{ADT2} will be overlapped in time. Thus, real T_{ADT}

will be less than the sum of the two periods.

C) *Redundancy and collision management*

To mitigate the effects of channel errors and collisions, the scheme uses repetition. Redundancy by repetition is ensured by setting a T_{listen} period long enough to ensure the repetition of several poll events from each node (ADV_SCAN_IND), and by the repetition of the adv^{seqADT} event, in addition to the inherent redundancy of the repetition in the three advertising frequency channels defined in BLE. Furthermore, a random time is budgeted before the transmission of the first adv^{seqADT} event to reduce the probability of collision between different relays.

Moreover, the application of a mechanism for resolution of collisions between nodes is required. This is associated with the *scannable undirected advertising events* illustrated in Fig. 4. The collision problem appears when two or more intermediate nodes, which are located at a similar distance from the source node, are involved in the T_{listen} period simultaneously, scanning in the same frequency channel (i.e. CH 37, 38, or 39) and sharing one or more neighbor nodes in poll mode. This could be the case of nodes 4 and 5 that are polled by node 6 in Fig. 5.

Nodes 4 and 5 respond simultaneously by sending the SCAN_REQ, which collide with each other, except if the capture effect allows suppression of the weaker signal at the receiver node. In the *scannable undirected advertising event*, this type of collisions is identified exclusively by the unsuccessful reception of the expected SCAN_RSP PDU after a time $T_{IFS.}$ In the absence of capture effect, collisions are persistent. Thus, a backoff algorithm needs to be applied to avoid or minimize the collisions between SCAN_REQ. A simple implementation was defined in BLE v4.2 [39]. Since BLE v5.0, the implementation is out of standardization, although the mechanism defined in BLE v4.2 remains suggested as an example and most real BLE chipsets implement it. The backoff procedure defined in v4.2 is simple. It uses two parameters: *backoffCount* and *upperLimit*. Upon entering the scan state, the *upperLimit* and *backoffCount* are set to one by the nodes. On every received ADV_SCAN_IND PDU that is allowed by the scanner filter policy, the *backoffCount* is reduced by one (until zero) and the SCAN_REQ PDU is only sent by the scanner when *backoffCount* becomes 0. If after sending a SCAN_REQ PDU a valid SCAN_RSP PDU is not received at the Link Layer of the scanner, it is considered a failure; otherwise, it is considered a success. The *upperLimit* value is doubled every two consecutive failures (until it reaches the value of 256), and halved (until it reaches the value of 1) after every two consecutive successes. Moreover, after every success or failure, the link layer selects a pseudo-random integer value for *backoffCount,* between one and *upperLimit* inclusive. As seen, the algorithm is simple, but the efficiency is severely affected by several drawbacks. 1) High probability of unnecessary backoff activation due to the lack of efficient discrimination between real collisions between SCAN_REQ and erroneous receptions of SCAN_REQ or SCAN_RSP due to fading or collisions with other PDU types. 2) Unfairness performance. A detailed analysis of backoff for BLE Neighbor Discovery Process (NDP), including a critical analysis of the impact of non-idealities of devices and the factors involved in the process, referred above, can be seen in [40]. That work also includes a proposal to improve the discovery latencies in high-density scenarios, where a large number of devices need to be discovered in very short periods of time by a scanner.

The use context of PSM-DMO is a planned deployment where a limited number of relay neighbor nodes is expected. If the number of relay neighbor nodes or other external interfering nodes is high, collisions increases and a re-adjustment of parameters is needed, being possible that the proposal loses its advantages and the system will enter in the standard operation. In this expected use context, in order not to unnecessarily increase the delay and, thus, the probability of detection within T_{listen} period, the maximum *upperLimit* should be greatly reduced, while a new *upperLimit* adjustment rule may be applied. In addition, some improvements, also included in [40], can be applied in the management of the procedure. Scanners in backoff state may be able to opportunistically monitor SCAN_RSP packets sent in response to the SCAN_REQ of another scanner (no matter the scanner could have sent the SCAN_REQ but backoff prevented it, or the scanner has not detected the ADV_SCAN_IND). For example, in Fig. 5, node 4 is in backoff state when node 6 sends the second ADV_SCAN_IND, which is also received by node 5. If node 4 receives de ADV_SCAN_IND, it does not transmit SCAN_REQ but waits for the SCAN_RSP. When another node (node 5) sends a SCAN_REQ, if node 4 receives a SCAN_RSP containing the identifier/address of the process it should send, it can assume that it sent the SCAN_REQ and continue with the adv^{seqADT} broadcast procedure.

In any case, note that in real scenarios, the capture effect allows detecting, in most cases, one of the SCAN_REQs that overlap. Therefore, collisions are very limited. According to [41], with a signal difference of 6–8 dB, the capture effect allows decoding the signal correctly. On the other hand, it is feasible for the nodes to scan at different frequencies, which implies that SCAN_REQ transmissions occur at different frequencies, without a collision.

Finally, note that the probability of exceeding the $TIMEOUTRX_{seqADT,}$ defined previously is significantly reduced after introducing this improvement.

D) *Other anomalies management*

The proposed mechanism needs to guarantee or maximize the probability that nodes that move to the scan mode are those that guarantee to reach the destination. Ideally, in absence of errors and collisions, in a T_{poll} period, a node correctly receives the ADV_SCAN_IND and, subsequently, the corresponding SCAN_RSP of all its neighbors, which then pass to scan mode. Nevertheless, in real conditions (where transmissions of ADV_SCAN_IND, SCAN_REQ, and SCAN_RSP are affected by collisions and errors), it is possible that only a portion of neighbors could correctly interact with the node and go into scan mode. In this case, it is not guaranteed that these neighbor nodes are the ones that allow reaching the destination of the data to be transmitted/retransmitted through the network.

In order to ensure that the destination could be reached, each node should know the number of neighbors and verify that all of them have passed to scan mode. However, in a scenario with dynamic topology, it should not be assumed that the list of neighbors is available. On the other hand, since detection is only limited by collisions, we opt for a mechanism that probabilistically guarantees that these nodes go into scan mode. In short, T_{listen} may include several T_{poll} intervals (N_{PO} greater than unity). As an improvement, N_{PO} can be chosen by applying an adaptive scheme. N_{PO} can start with a high value and, as time passes, if nodes are efficiently detected in a short time, N_{PO} can be decreased. The minimum value of N_{PO} must be high enough so that there are no problems if something changes in the topology.

4. Energy consumption

Once the basis of the proposal has been presented, in this section we analyze the different consumption of the devices with or without applying the PSM-DMO proposal.

First, when working in standard mode, i.e., without applying the PSM-DMO proposal, a device in standby (without transmissions) but scanning continuously has a daily energy consumption ($E_{stb,woTX}$) in Joules of:

$$E_{std,woTX} = \mathrm{V}\cdot \mathrm{I_{scan}}\cdot \mathrm{T}_{DAY} \tag{11}$$

where V is the working voltage, T_{DAY} is the duration of a day, and I_{scan} is the current consumption of the device under analysis while scanning.

Then, to estimate the energy consumption of PSM-DMO proposal with (E_{wTX}) and without transmissions (E_{woTX}) and the standard mode with transmissions ($E_{stb,wTX}$), we first estimate the energy consumption associated with *scannable undirected advertising events* (named

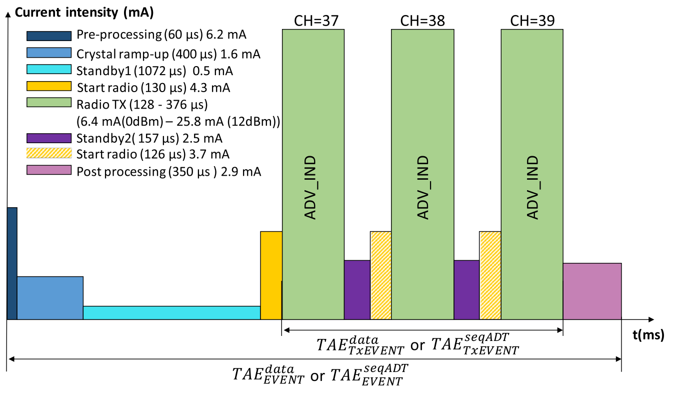


Fig. 9. Example of the current consumption of non-connectable undirected advertising events [36].

$eTAE^{Poll}_{EVENT}$), and non-connectable undirected advertising events used in adv^{seqADT} sequence (named $eTAE^{seqADT}_{EVENT}$) and data transmission (named $eTAE^{data,TX}_{EVENT}$).

In all these events, periods linked to the transmission of the ADV_IND or ADV_SCAN_IND, SCAN_REQ, SCAN-RSP packets are combined with scan periods and other processes (e.g. pre-processing, crystal ramp-up, etc.) with different current intensity levels. The objective is to calculate the global energy consumption associated with the whole event as the sum of energy consumption of all of them. Voltage, current intensity (including I_{scan} and I_{idle}), and durations can be found in the datasheets of the chipset manufacturers. For example, in [36] or [42]. Fig. 9 shows the current intensity (mA) of non-connectable undirected advertising events, extracted from [36] and validated experimentally using the nRF52840 DK.

In the case of $eTAE^{Poll}_{EVENT}$, it is clear that the consumption is not the same for the node in scan mode and the node that polls. Since the consumption linked to a full event (up to the reception of SCAN_RSP) is always higher for the node in polling mode (up to transmission of SCAN_RSP), consumption associated with this node could be used as a threshold approach for both. In any case, it is necessary to distinguish between several situations. We consider the two with higher probability. 1) The event implies a response (a SCAN_REQ is received and a SCAN_RSP is sent). We name it $eTAE^{Poll(wR)}_{EVENT}$. 2) No response is received, named $eTAE^{Poll(woR)}_{EVENT}$. Fig. 10 shows an example of current consumption in the case of no-response (a) and in the case of receiving a response on channel 37 (b). Indeed, it is feasible to obtain a response in only one of the channels (37, 38, or 39), in two of them, or all three. Probability depends on the fact that a hop away from the polling node, there are more than two or three nodes, respectively, in the T_{listen} period. It also depends on the probability that they are scanning at different frequencies. However, in this paper, from the point of view of consumption, the case where there is only a response in one of the channels will be taken into account. Finally, duration (named $TAE^{Poll(wR)}_{EVENT}$) and energy consumption ($eTAE^{Poll(wR)}_{EVENT}$) are the same if the response is received on channels 37 and 38 or if it is received on the last one (CH = 39).

Once $eTAE^{Poll(wR)}_{EVENT}$, $eTAE^{Poll(woR)}_{EVENT}$, $eTAE^{seqADT}_{EVENT}$, and $eTAE^{data,TX}_{EVENT}$ are obtained, the objective is to compare E_{wTX} and $E_{stb,wTX}$.

If N_{TX} transmission events per day are considered, each one of them sending N_{data} packets and, assuming $N^{data}_{TC} = R^{data}_{RC}$ the consumed energy following the standard BLE mesh specifications ($E_{stb,wTX}$), is computed with Eq. (12):

$$E_{std,wTX} = N_{TX}\cdot N_{data}\cdot\left(P^{data}_{RC}+1\right)\cdot\left(N^{data}_{TC}+1\right)\cdot eTAE^{data}_{EVENT}+ \\ +V\cdot I_{scan}\cdot\left(T_{DAY}-N_{TX}\cdot N_{data}\cdot\left(P^{data}_{RC}+1\right)\cdot\left(N^{data}_{TC}+1\right)\cdot TAE^{data}_{EVENT}\right) \quad (12)$$

Realize that the flow of messages linked to the standard mesh only includes T_{DATA} flow illustrated in Fig. 5. To compute $E_{stb,wTX}$, E_{wTX}, and $E_{woTX,}$ note that the duration of the events ($TAE^{Poll(woR)}_{EVENT}$,$TAE^{Poll(wR)}_{EVENT}$, TAE^{seqADT}_{EVENT} and TAE^{data}_{EVENT}) include the preprocessing times, crystal increment, standby1, radio start, and transmission post-processing (see Figs. 9 and 10). Nevertheless, these values are not included in the delay analysis in Eqs. (7) and (8) as they overlap with the minimum times between consecutive events and random times that the relays apply before retransmissions. For delay analysis purposes, $TAE^{Poll(woR)}_{TxEVENT}$, $TAE^{Poll(wR)}_{EVENT}$, $TAE^{seqADT}_{TxEVENT}$ and $TAE^{data}_{TxEVENT}$ are used.

The energy consumption of PSM-DMO proposal in a day without data transmissions (E_{woTX}) is computed using Eq. (13):

$$E_{woTX} = \lceil T_{DAY}/T_{Poll}\rceil\cdot eTAE^{Poll(woR)}_{EVENT}+ \\ +V\cdot I_{dle}\cdot\left[T_{DAY}-\lceil T_{DAY}/T_{Poll}\rceil\cdot TAE^{Poll(woR)}_{EVENT}\right] \quad (13)$$

where $\lceil T_{DAY}/T_{Poll}\rceil\cdot eTAE^{Poll(woR)}_{EVENT}$ is the extra energy consumption linked to the poll process, while the difference between $E_{stb,woTX}$ computed in (11) and $E_{woTX,}$ computed in (13) is the net energy savings linked to the proposal.

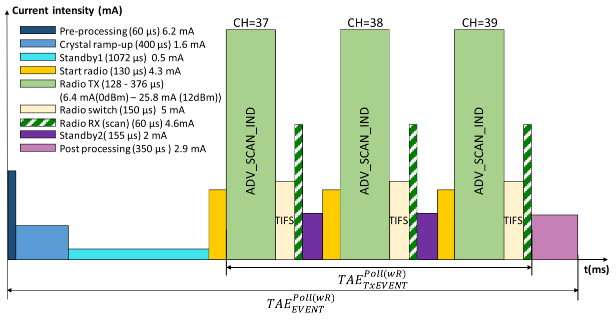
For the PSM-DMO proposal, if N_{TX} transmission events per day are considered, the consumed energy (E_{wTX}) can be divided into the three components showed in Eq. (14): energy consumption in idle state (E_{idle}), consumption linked to periodic polling events when the node is in PSM (E_{poll}) and consumption when the node is involved in a transmission ($E_{activePeriod}$). The calculation is made for the source node since it represents an upper bound.

$$E_{wTX} = E_{poll} + E_{idle} + E_{activePeriod} \quad (14)$$

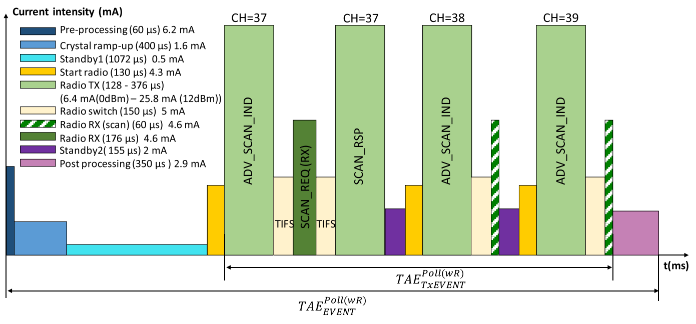
E_{poll} is obtained using Eq. (15):

$$E_{poll} = \lceil\left[T_{DAY} - N_{TX}\cdot T_{sourceADT}\right]/T_{Poll}\rceil\cdot eTAE^{Poll(woR)}_{EVENT} \quad (15)$$

with:



(a) Without response



(b) With response in channel 37

Fig. 10. Example of the current consumption of scannable undirected advertising events [36].

$$T_{sourceADT} = \left(N_{PO} \cdot \left(T_{poll} + 10\right) + T_{seqADT} + T_{ADT}\right) \tag{16}$$

being T_{ADT} computed using Eqs. (2), (5), and (6).

$E_{activePeriod}$ is obtained using Eq. (17):

$$E_{activePeriod} = E^{TX}_{activePeriod} + E^{scan}_{activePeriod} \tag{17}$$

where $E^{TX}_{activePeriod}$, obtained with Eq. (18), computes the consumption linked to transmissions:

$$E^{TX}_{activePeriod} = N_{TX} \cdot \left\{ \begin{array}{c} n_{neighPoll} \cdot eTAE^{Poll(wR)}_{EVENT} + \\ +\left(N^{seqADT}_{TC} + 1\right) \cdot eTAE^{seqADT}_{EVENT} + \\ +N_{data} \cdot \left(P^{data}_{RC} + 1\right) \cdot \left(N^{data}_{TC} + 1\right) \cdot eTAE^{data}_{EVENT} \end{array} \right\} \tag{18}$$

and$E^{scan}_{activePeriod}$, obtained with Eqs. (19) and (20), computes the consumption linked to the scan periods.

$$E^{scan}_{activePeriod} = V \cdot I_{scan} \cdot \left(T_{sourceADT} - T^{TX}_{activePeriod}\right) \tag{19}$$

$$T^{TX}_{activePeriod} = N_{TX} \cdot \left\{ \begin{array}{c} n_{neighPoll} \cdot TAE^{Poll(wR)}_{EVENT} + \\ +\left(N^{seqADT}_{TC} + 1\right) \cdot TAE^{seqADT}_{EVENT} + \\ +N_{data} \cdot \left(P^{data}_{RC} + 1\right) \cdot \left(N^{data}_{TC} + 1\right) \cdot TAE^{data}_{EVENT} \end{array} \right\} \tag{20}$$

Note that in the calculation of both $E^{TX}_{activePeriod}$ and $E^{scan}_{activePeriod}$, it was assumed that $n_{neighPoll}$ poll events are required in average in a node as the sum of two situations: the periods where a node polls an active scanner and then, the T_{listen} period where this node switches to the active mode.

Finally, E_{idle} is obtained using Eq. (21):

$$E_{idle} = V \cdot I_{dle} \cdot \left\{ \begin{array}{c} T_{DAY} - N_{TX} \cdot T_{sourceADT} - \\ -\lceil [T_{DAY} - N_{TX} \cdot T_{sourceADT}] / T_{Poll} \rceil \cdot TAE^{Poll(woR)}_{EVENT} \end{array} \right\} \tag{21}$$

5. Parameter selection and results

The most important design parameters of the PSM-DMO proposal refer to the configuration of the polling process (specifically T_{poll} and N_{PO}) and the transmission of the sequence of ADT messages ($netInt^{seqADT}$and N^{seqADT}_{TC}). The choice of these parameters is conditioned by the probability of non-detection of ADV_SCAN_IND, SCAN_REQ, SCAN_RSP, and ADV_IND, respectively. Also, the longer the T_{poll} time, the lower the power consumption, but this is at the cost of including an extra delay, not present in the standard mesh mode.

As a previous step to choosing these parameters, the duration of each type of PDU must be specified.

Fig. 4 shows the packet formats associated with *scannable undirected advertising events*, used in the period polling process. ADV_SCAN_IND and SCAN_RSP may include unsegmented control mesh packets with an Opcode of 3 bytes and P_{id} additional data in the SCAN_RSP. This implies a radio transmission time of $T_{ADVpoll}$ =*312* μs, $T_{scanRSPpoll}$ = 336 μs and $T_{scanREQpoll}$ = 176 μs.

In the adv^{seqNAV} sequence, the ADV_IND packets must transport an unsegmented control *PDU* with maximum size *(*T_{ADT1} (3 bytes) and T_{ADT2} (3 bytes)*,* TTL_{ADT} (1 byte)*,* and *PId* (3bytes)). Thus, $T_{ADVseqADT}$=376 μs.

For data transmission itself, ADV PDUs of maximum duration ($T_{ADVdata}$ =376 μs) are considered.

5.1. Parameter selection of ADT sequence

Given a configuration of the time between adv^{seqADT} events, the choice of the number of events is fixed to ensure a threshold for the collision probability of the ADV_IND packets associated with the *non-connectable and non-scannable events.*

Some considerations are made to simplify the collision probability calculation. First, nodes involved in the reception of adv^{seqADT} events have, with high probability, interfering neighbor nodes sending the adv^{seqADT} or only the ADV_SCAN_IND linked to the polling process. See, for instance, node 6 in Fig. 5 that potentially receives the adv^{seqADT} from nodes 4 and 5 and an ADV_SCAN_IND from node 8. The advertising interval between consecutive poll events is expected to be higher than between adv^{seqADT}. Besides, $T_{ADVseqADT}$ is slightly longer than $T_{ADVpoll}$. Thus, the impact of polling on the collision probability is lower than those of ADT transmission. That is, collision probability may be calculated assuming that all the neighbors transmit adv^{seqADT} events, being this probability an upper bound of a real scenario. The collision probability between ADV_IND is a stochastic process with memory. However, assuming continuous advertising, in a long term and averaged analysis we can simplify the calculation.

The time between the start of two consecutive advertising events (affected by the imposed timing restrictions within the event) is controlled by the *advInterval* parameter ($T_{advInterval}$≥20 ms in BLE mesh) plus a random variable ($\tau_{advDelay}$) between 0 and $T_{advDelayMax}$=10 ms ($rand_{10}$) and, the transmission time of the ADV PDU (T_{advIND}) is fixed (in this case $T_{ADVseqADT}$). The probability that transmission from a reference node (started in a time instant *t*) collides with another one in each frequency of the event, corresponds with the probability that one neighbor node starts its own transmission in the interval $[t - T_{advIND}, t + T_{advIND}]$. As the mean time between consecutive advertisement transmissions is $T_{advInterval} + \overline{\tau_{advDelay}}$, the probability of collision is$2 \cdot T_{advIND} / (T_{advInterval} + \overline{\tau_{advDelay}})$. Finally, if transmissions of theN_{BLE} nodes are independent, the non-detection probability due collision ($P^{col}_{NDAdvIND}$) in a scenario with N_{BLE} interfering nodes is obtained with Eq. (22):

$$P^{col}_{NDAdvIND} = 1 - \left(1 - \frac{2 \cdot T_{advIND}}{T_{advInterval} + \overline{\tau_{advDelay}}}\right)^{N_{BLE}-1} \tag{22}$$

Nevertheless, in the PSM-DMO scenario, it could be possible for the nodes to synchronously end the T_{listen} period. In this case, the time difference among the respective advertisement events is the sum of two random variables ($\tau_{advDelay}$ and $rand_{seqADT}$ defined in Section 3) and confined to the time interval [0, max($rand_{seqADT}$)+10 ms]. Note that mesh specification sets that, upon receiving a *network PDU* at the Advertising Bearer Network Interface, it shall transmit it using the value of the *Network Transmit state.* This includes N_{TIS} and the $rand_{10}$ requirement between each transmission, while $rand_{10}$ also affects the first transmission [15]. Assuming that, the non-detection probability is obtained with Eq. (23):

$$P^{col}_{NDAdvIND} = 1 - \left(1 - \frac{2 \cdot T_{advIND}}{\max\left(rand_{seqADT}\right) + 10}\right)^{N_{BLE}-1} \tag{23}$$

Finally, considering the channel effects through BLER, the probability of non-detection is obtained with Eq. (24):

$$P^{col+BLER}_{NDAdvIND} = P^{col}_{NDAdvIND} + \left(1 - P^{col}_{NDAdvIND}\right) \cdot BLER \tag{24}$$

In the PSM-DMO deployment scenarios, the number of neighboring nodes is expected to be low. For example, we could assume a mean number not higher than 4, and still being a rather pessimistic situation. On the other hand, a main requirement is to reduce the total delay, although, it exists a compromise between reducing the time and the number of required adv^{seqADT} events. Reducing the time increases the collision probability and, consequently, more transmissions are needed.

Under these conditions, considering a worst-case with 4 neighbor nodes transmitting simultaneously, T_{advIND} =$T_{ADVseqADT}$=376 μs ($TAE^{seqADT}_{TxEVENT}$=1.694 ms), the minimum *advInterval* (20 ms) and $rand_{10}$, the collision probability (or non-detection probability) is 8.8% considering Eq. (22) (note that, Eq. (23) provides also a low collision probability, even considering a minimum value of the max($rand_{seqADT}$)=20 ms).

On the other hand, it could be expected a non-negligible BLER. ADV packets do not include channel coding. Therefore, for receiver sensitivity values, bit error rates $=10^{-3}$ are expected, which translate to BLER around 30%. However, this assumption is unreal in most cases. When choosing this type of solution, deployments should be more or less controlled. Coverages could be planned to receive signal levels high enough to achieve must better BLER values (for example, below 10%), even for the largest packages (376 µs). With BLE= 10%, non-detection probability is around 17.9%. Under these conditions, considering 3 consecutive events, non-detection probability due to collisions and BLER is reduced to 0.57% (to 0.0033% with 6). As defined in Section 2; mesh mode allows the nodes to receive multiple copies of the original packet and repetitions, controlled by the parameters P_{RC}, N_{TC}, *and* R_{RC}. In this case, *TTL=0* (TTL_{seq}=0) is applied and we can consider a reasonable configuration *advInterval*=$netInt^{seqADT}$=20 ms, $P_{RC}^{seqADT}=0$ and N_{TC}^{seqADT}=2 to budget 3 adv^{seqADT} events for BLER=10%. In any case, the non-detection probability remains clearly overestimated. It does not take into account the capture effect. BLER is often much better. On the other hand, in case of no detection of any *ADT* packets after a $TIMEOUTRX_{seqADT}$, the receiver node reactivates the polling process. Upon receipt of a poll packet by a previously detected node, nodes could resent the adv^{seqADT} sequence again. However, this situation is highly unlikely.

5.2. Parameter selection of data transmissions

Concerning data transmission, the same considerations can be made for the adv^{seqADT} sequence. Assuming $T_{ADVdata}$ =376 µs ($TAE_{TxEVENT}^{data}$=1.694 ms), we can consider a reasonable configuration *advInterval*=20 ms and 3 repetitions for BLER=10% in each hop. The managed flooding protocol applied in the mesh mode provides an implicit redundancy, controlled by the parameters P_{RC}, N_{TC}, and R_{RC}. Furthermore, data energy consumption computes equally in both the standard mode and in the saving proposal. Thus, an arbitrary configuration can be applied. In any case, it must be a reasonable configuration from the point of view of non-detection probability. We propose a more redundant configuration with $pubInt^{data}=100$ ms, $P_{RC}^{data}=1$, $netInt^{data}=20$ ms, $N_{TC}^{data}=2$, $relInt^{data}=20$ ms and $R_{RC}^{data}=2$. Alternatively,$P_{RC}^{data}=0$, $netInt^{data}=20$ ms, $N_{TC}^{data}=5$, $relInt^{data}=20$ ms and $R_{RC}^{data}=5$. This configuration implies the retransmission of each original packet a total of 6 times.

5.3. Parameter selection of interrogation process

The choice of configuration parameters linked to the poll process (T_{poll} and N_{PO}) depends on the non-detection probability of the ADV_SCAN_IND, SCAN_REQ, and SCAN_RSP. It is required to choose T_{poll} and N_{PO}, to obtain a tradeoff between delay and energy consumption always ensuring reliability.

First, the most relevant issue is determining the minimum T_{poll} in order to limit the non-detection probability of SCAN_RSP PDU ($P_{NDScanRSP}$) in a given frequency.

Some considerations are made in order to simplify the collision probability calculation. We consider only one node in T_{listen} period and a number N_{BLE} of interfering nodes, all of them involved in the interrogation process (see, for instance, nodes 2, 3, 4, and 5 in Fig. 5 and source node), Assuming this, in absence of channel errors, non-detection probability of the ADV_SCAN_IND ($P_{NDScanRSP}$) depends on the probability of collision with other ADV_SCAN_IND ($P_{NDAdvIND}^{col}$). This collision probability can be obtained using Eq. (22). But, $P_{NDAdvIND}$ also depends on the probability that the receiver (node in T_{listen} period) is already involved in a signaling processing period and decoding gaps ($P_{NDAdvIND}^{sigproc+gap}$), according to Eq. (25).

$$P_{NDAdvIND}=P_{NDAdvIND}^{col}+\left(1-P_{NDAdvIND}^{col}\right)\cdot P_{NDAdvIND}^{sigproc+gap} \tag{25}$$

The signal processing period is defined as the time interval needed to complete the event. It always includes the $T_{IFS}+T_{ScanREQ}+T_{IFS}$ interval and a variable time which depends on the successful/unsuccessful transmission of the SCAN_REQ PDU and thus, the subsequent transmission of a SCAN_RSP. In [37] a specific model is presented to characterize this probability in devices that reproduce the specification and in real chipsets. This model is used as a reference for the estimates made here.

Using the model described in [37], the non-detection probability due to the collisions of the SCAN_REQ ($P_{NDScanREQ}^{col}$) and SCAN_RSP ($P_{NDScanRSP}^{col}$) packet are obtained with Eqs. (26) and (27):

$$P_{NDScanREQ}^{col}=1-\left(1-\frac{\min(T_{IFS},T_{advIND})+T_{scanREQ}}{T_{advEvent}}\right)^{N_{BLE}-1} \tag{26}$$

$$P_{NDScanRSP}^{col}=1-\left(1-\frac{\min(T_{IFS},T_{advIND})+T_{scanRSP}}{T_{advEvent}}\right)^{N_{BLE}-1} \tag{27}$$

From there, the non-detection probability of SCAN_REQ ($P_{NDScanREQ}$) and SCAN_RSP ($P_{NDScanRSP}$) is obtained according to the expressions (28) and (29), respectively [37]:

$$P_{NDScanREQ}=1-(1-P_{NDAdvIND})\cdot\left(1-P_{NDScanREQ}^{col}\right) \tag{28}$$

$$P_{NDScanRSP}=1-(1-P_{NDAdvIND})\cdot\left(1-P_{NDScanREQ}^{col}\right)\cdot\left(1-P_{NDScanRSP}^{col}\right) \tag{29}$$

To include BLER, it is only needed to replace in Eqs. (28) and (29) $P_{NDAdvIND}$, $P_{NDScanRSP}^{col}$ and $P_{NDScanRSP}^{col}$ by the following expressions (30), (31), and (32).

$$P_{NDAdvIND}^{+BLER}=P_{NDAdvIND}+(1-P_{NDAdvIND})\cdot BLER \tag{30}$$

$$P_{NDScanREQ}^{col+BLER}=P_{NDScanREQ}^{col}+\left(1-P_{NDScanREQ}^{col}\right)\cdot BLER \tag{31}$$

$$P_{NDScanRSP}^{col+BLER}=P_{NDScanRSP}^{col}+\left(1-P_{NDScanRSP}^{col}\right)\cdot BLER \tag{32}$$

If two or more nodes are involved in the T_{listen} period and they are scanning at the same frequency, a collision occurs in SCAN_REQ with 100% probability if the capture effect is obviated. The probability of this event is considered separately and is solved using a backoff algorithm according to the guidelines mentioned in Section 3.

To compensate collisions among SCAN_REQ and backoff activation in two or more nodes (e.g. nodes 4 and 5 polled by node 6 in Fig. 5), we consider that at least three poll events are required in the T_{listen} period. Therefore, assuming the previous analysis as a valid approximation, Table 2 shows the non-detection probabilities of SCAN_RSP for different T_{poll} values, in ideal conditions and for BLER = 10%, in the case of a number of neighbors (relays) equal to 4. In a planned deployment, which is the ideal scenario of application of the PSM-DMO and BMADS, it is reasonable that there are a limited number of relay neighbors. If there are additional interfering networks, collisions increase and a readjustment of parameters will be required. However, in many cases, PSM-DMO will still be better than the standard as long as the overall number of transmissions and/or the number of hops in the planned networks are moderate. PSM-DMO is defined as an optional operation mode, adaptively selectable. If, because of an increased traffic or interference, depending on the number of hops, PSM-DMO loses its benefits, the system must enter in the standard operation.

Table 2 also includes the mean required N_{PO} and rounded bound of N_{PO} in order to achieve a non-detection probability lower than 1% in the T_{listen} period. As it can be seen, by increasing T_{poll}, the probability of non-detection decreases, but, in all cases, BLER is the parameter with the most severe impact. Under ideal conditions, assuming 3 polling events in the T_{listen} period ($N_{PO}=3$) would be enough even the smallest T_{poll} value. However, if BLER=10%, a double number of events are required.

Recall that once a node receives a SCAN_REQ it sends the SCAN_RSP, disables the polling process, and proceeds to passive continuous scan.

Table 2
$P_{NDScanRSP}$ variation with T_{poll} and BLER for four competing nodes and one node in T_{listen} period. Mean N_{PO}, upper bound of N_{PO} required to achieve a non-detection probability lower than 1% and estimated $n_{neighPoll}$.

	BLER=0%				**BLER=10%**			
T_{poll}	$P_{NDScanRSP}$	$MeanN_{PO}$	N_{PO}	$n_{neighPoll}$	$P_{NDScanRSP}$	$MeanN_{PO}$	N_{PO}	$n_{neighPoll}$
20 ms	0.2245	3.09	4	7	0.4347	5.53	6	9
30 ms	0.1676	2.58	3	6	0.3932	4.94	5	8
40 ms	0.1337	2.29	3	6	0.3685	4.61	5	8
50 ms	0.1112	2.1	3	6	0.3521	4.41	5	8
100 ms	0.0604	1.64	2	6	0.315	3.98	4	8
200 ms	0.0316	1.34	2	6	0.294	3.76	4	7
300 ms	0.0213	1.20	2	6	0.2866	3.68	4	7
400 ms	0.0161	1.12	2	6	0.2828	3.65	4	7
500 ms	0.0130	1.06	2	6	0.2805	3.62	4	7
1 s	0.0065	0.92	1	5	0.2758	3.57	4	7

Table 3
Parameters used in the evaluation.

Common parameters			
Current consumption at 8 dBm	16.4 mA		
Current consumption receiving (scan)	4.60 mA		
Current consumption in idle mode	0.01 mA		
Working voltage	3 V		
Processing times T_{PROC}^{relay} and T_{PROC}^{sink}	20 µs		
$rand_{10}$	10 ms		
Scanned channels	37, 38 and 39		
Data parameters			
	PSM-DMO	**BMADS**	
Publish retransmit count, P_{RC}^{data}	0	0	
Publish retransmit interval, $pubInt^{data}$	N/A	N/A	
Network transmit count, N_{TC}^{data}	5	k-1=[2–5]	
Network transmit interval, $netInt^{data}$	20 ms	20 ms	
Relay retransmit count, R_{RC}^{data}	5	k-1=[2–5]	
Relay retransmit interval, $relInt^{data}$	20 ms	20 ms	
Time of advertising event, $TAE_{TxEVENT}^{data}$	1.694 ms	1.694 ms	
$rand_{MAX=}$*max (rand) at the relay nodes*	20 ms	20 ms	
PSM-DMO parameters			
Poll process		**ADT sequence**	
$T_{advIntervall}$= T_{poll}	[20–1000]s	TTL_{seq}	0
Max ($rand_{10}$)	10 ms	P_{RC}^{seqADT}	0
N_{PO}	1, 3–6	N_{TC}^{seqADT}	2–5
$T_{ADVpoll}$	312 µs	$netInt^{seqADT}$	20 ms
$T_{scanREQpoll}$	176 µs	$TAE_{TxEVENT}^{seqADT}$	1.694 ms
$T_{scanRSPpoll}$	336 µs	$rand_{MAX}$= =max($rand_{seqADT}$)	20 ms
BMADS parameters			
P_{RC}^{seq}	[0–7] in steps of 1	R_{RC}^{seq}	N/A
$pubInt^{seq}$	[50, 1600] ms in steps of 50	$relInt^{seq}$	N/A
N_{TC}^{seq}	[0–7] in steps of 1	*Redundancy level, k*	1, 3–6
$pubInt^{seq}$	[20, 320] ms in steps of 10	$TAE_{TxEVENT}^{seq}$	1.694 ms
$scanWin^{seq}$ $scanInt^{seq}$	Chosen together with P_{RC}^{seq} $pubInt^{seq}$ N_{TC}^{seq} and $pubInt^{seq}$ according with k.	TTL_{seq}	0

Thus, the number of competing nodes decreases along the T_{listen} and they are affected by lower collision probability. That is, if there are several nodes (for example, the 4 used for the estimation), the values of the collision probability included in Table 2 are really upper bounds.

On the other hand, if several nodes interrogate the same node in T_{listen} period, as long as a SCAN_REQ was successfully sent, the reception of at least one SCAN_RSP from one of these nodes activates the sending of the adv^{seqADT}. So, the real effects of collisions and BLER are less significant than the estimated in Table 2. On the contrary, when there is a single node, the expected BLER is a key parameter.

Concerning the $n_{neighPoll}$ parameter used in Eqs. (18) and (20) to compute consumption energy, $n_{neighPoll}$ is the average number of poll events in which a node is involved. It is selected in such a way that it represents a conservative bound. $n_{neighPoll}$ is the sum of two terms: 1) the mean number of poll events that the node transmits during the period it is being scanned by a neighbor in the T_{listen} period and, 2) the average number of poll events in which it is involved (responding with poll scan request messages to their neighbors) when it passes to the T_{listen} period. It is calculated based on the $P_{NDScanRSP,}$,using the Eq. (33) where n_{neigh} is the number of estimated neighbors. Realize that to obtain an upper bound of the consumption, as a simplified approach, poll events are considered completed (up to the reception of SCAN RSP transmission) and the energy linked to the polling node. Table 2, includes the corresponding values for the derived N_{PO} when n_{neigh}=4.

$$n_{neighPol_l} = \left(n_{neigh} + 1\right)\cdot\left(1 - P_{NDScanRSP}\right)\sum_{n=1}^{N_{PO}} n\left(P_{NDScanRSP}\right)^{n-1} \tag{33}$$

With these assumptions, the performance of PSM-DMO will be evaluated in terms of energy consumption and delay for different T_{poll} values and N_{PO} between 3 and 6, always assuming n_{neig}=4 and a worst-case of $n_{neighPoll}$=9.

5.4. Discussion and results

To evaluate the achievable gains of the PSM-DMO proposal, the energy consumption of the network has been computed and been expressed relative to the standard mode operation of the Bluetooth Mesh network. That is, it is computed as a percentage of the continuous scan option. Energy consumption is computed from the source node point of view related to the passive and active scan intervals. Nevertheless, $n_{neighPoll}$ is computed as indicated in (Eq. (33), same as if it were a relay.

The energy consumption is lower for relay and sink nodes, being the differences very significant when the size of the network grows. Although, in reality, any node can be a source, relay or sink, the energy consumption is always calculated for the worst node (the one that acts as a source) to evaluate the benefits of PSM-DMO. In addition, it is clear that PSM-DMO introduces an additional delay in the start of data transmission in mesh mode. The delay is computed in the source as the time elapsed from the data generation event to the beginning of data transmission. This delay is an upper bound of the time it takes for all nodes to be activated. A main requirement of the configuration is to reduce the total delay, although it is expected that a compromise exists between reducing the time and the energy consumption.

In any case, recall that the proposal is defined as an optional operation mode, eligible adaptively when nodes operate in scenarios with applications that are delay tolerant and/or make infrequent transmissions. In other cases, standard mode operation remains active. If the system operates in conditions where traffic is heavy and the number of hops required is also high, it should operate in normal operating mode.

Evaluation has been performed for different T_{poll} and N_{PO} values, in several scenarios that differ in their traffic volume, measured in the number of transmission events (TX) per time unit (a number that we translate in the number of transmissions event per day, N_{TX}) and in the network size. The number of relays is not limited in the deployment but, according to the specification, the number of hops is limited by the TTL. Thus, we define the networks size in terms of the number of hops required in each transmission to reach the destination and equal to the TTL (TTL=TTL_{ADT}). That is, if TTL is set to the maximum 127, the maximum number of intermediate relays for a transmission is 126. Parameters used in the evaluation are summarized in Table 3.

The proposal does not impose any restriction on the valid topology of the network. But, recall that parameter selection has been performed according to some expected characteristics of the deployments and about the density of the neighbors.

Power consumption and delay have been realistically considered. In particular:

- The consumption data are extracted from [36] and validated experimentally using the nRF52840 DK. Data correspond with that represented Fig. 9 and Fig. 10, I_{dle}=0.01 mA and I_{scan}=4.6 mA and I_{TX}=16.4 mA (8 dBm).
- The processing time at relay and sink nodes was also measured from the test-bed [43].
- The rest of the parameters are taken from the range of values defined BLE mesh specifications (verified with real measurements). Selected values correspond with those chosen in the parameter selection.

Fig. 11 shows the relative energy consumption with respect to a standard Bluetooth Mesh operation for a different number of poll events in the T_{listen} period (N_{PO}) and time between consecutive poll events (T_{poll}). Evaluation is performed considering one transmission per day (1TX/day) and one transmission per minute (1TX/min), in scenarios with TTL_{ADT}=4, 10, 50, and 127 (from now named only as TTL). Unrepresented values are not possible options. For example, for 1 TX/min and TTL=127, settings with the time between polling events of 100 ms or more are not feasible. As expected, recall that for 1 TX/day relative energy consumption is similar for different values of TTL.

In the same conditions, Fig. 12 shows the extra delay introduced by

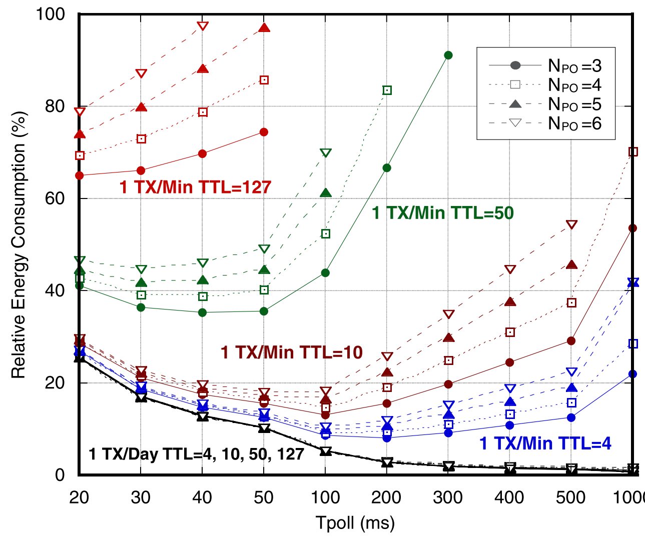


Fig. 11. Relative Energy Consumption (%) of PSM-DMO proposal compared with the standard mode implementation.

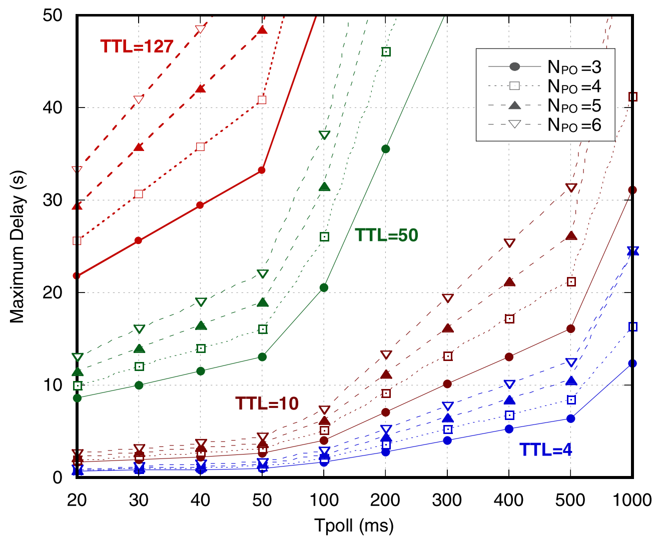


Fig. 12. Additional delay introduced by the PSM-DMO proposal compared with the standard mode implementation.

the PSM-DMO. Note that, in this case, the delay does not depend on the number of transmissions per unit of time. It only depends on T_{poll}, N_{PO}, and TTL.

For one TX/day, Table 4 shows the values illustrated in Fig. 11 for the extreme cases (TTL=4 and TTL=127 with N_{PO} =3 and N_{PO} =6), including also unrepresented delays in Fig. 12 for TTL=127. Power savings are higher than 99% in some configurations. In addition, Table 4 shows in column No TX the relative energy consumption of the proposal compared to a standard BLE mesh network in the absence of transmissions.

Now we evaluate simultaneously both energy consumption and delay.

When the number of transmission events is low (1TX /day) it is remarkable that the relative energy consumption reduces up to 25.5% for T_{poll}=20 ms, to 5.3% for T_{poll}=100 ms, and until 1.3% for T_{poll}=500 ms, with a negligible impact due to N_{PO} or TTL. However, the delay depends on the number of hops (TTL). As TTL grows, the transmissions might increase the delay. For instance, with TTL=4, the delay varies in the range of [0.67 s-1.03 s] for T_{poll}=20 ms, [1.63 s-2.95 s] for T_{poll}=100 ms and [6.43 s-12.55 s] for T_{poll}=500 ms, depending on the N_{PO} value. Really, taking into account N_{PO} requirements of Table 2 for a hostile and very improbable scenario with BLER=10%, delay is 1.03 s (N_{PO} =6 and T_{poll}=20 ms), 2.07 s (N_{PO} =4 and T_{poll}=100 ms), and 8.5 s (N_{PO} =4 and T_{poll}=500 ms). That is, results are very good both in terms of save of energy and delay. However, as TTL grows, the transmissions might increase its delay. In any case, there is a tradeoff between delay and energy saving. For instance, if TTL=127, a relative consumption of 1.2% or 1.4% can be achieved with T_{poll}=1 s and N_{PO} =3 and N_{PO} =4 but with a delay up to 6.6 min and 8.7 min, respectively. In the other extreme, a delay lower than 33.2 s can be achieved with T_{poll}=20 ms and N_{PO} =6 and a relative consumption of 25.5%. A tradeoff can be achieved, for instance, for T_{poll}=200 ms and N_{PO}=4. In this case, relative consumption is 5.4% with a delay of around 1 min (66 s).

Notice that for T_{poll}=1000 ms relative energy consumption can be reduced up to 0.75% for TTL=4 or TTL=10 and up to 1.63% for TTL=127. Thus, a node having a battery of 4400 mAh, the operational life would be extended from 40 days to 43.7 years (for TTL up to 10) or 6.7 years for the largest networks (TTL=127 hops). Consequently, we can conclude that the proposed mechanism is a remarkably interesting option to achieve power-efficient Bluetooth Mesh networks where traffic is infrequent and for delay-tolerant applications.

As traffic increases, for instance, 1 TX/min =1440TX/day (see Fig. 11), relative energy consumption grows and the mechanism

Table 4
Detail of Relative Energy Consumption (%) of PSM-DMO proposal compared with the standard mode implementation and Delay. 1 TX/day and without TX.

T_{poll}	NoTX	1 TX/ day					
		TTL=4		TTL=127			
		Energy		Energy	Delay	Energy	Delay
		N_{PO}=3	N_{PO}=6	N_{PO}=3	N_{PO}=3	N_{PO}=6	N_{PO}=6
20 ms	25.500	25.500	25.487	25.500	21.8 s	25.523	33.2 s
30 ms	17.100	17.100	17.065	17.100	25.6 s	17.112	40.8 s
40 ms	12.900	12.900	12.854	12.900	29.4 s	12.910	48.5 s
50 ms	10.300	10.300	10.327	10.400	33.2 s	10.393	56.1 s
100 ms	5.270	5.270	5.275	5.340	52.3 s	5.385	1.6 min
200 ms	2.740	2.750	2.751	2.860	1.5 min	2.947	2.8 min
300 ms	1.900	1.910	1.911	2.060	2.1 min	2.193	4.1 min
400 ms	1.480	1.490	1.493	1.680	2.8 min	1.860	5.4 min
500 ms	1.230	1.240	1.243	1.470	3.4 min	1.695	6.6 min
1 s	0.723	0.738	0.751	1.188	6.6 min	1.630	13.0 min

logically tends to a standard continuous scan mode, particularly when the number of hops grows above TTL=50 (i.e. TTL=127). However, the energy saving is appreciable even with a high number of hops (i.e. TTL=50). As shown in Fig. 11, not all the configurations are feasible. In addition, a higher time between polling events does not necessarily imply less energy consumption. Certainly, this reduction occurs when there are no transmissions or a reduced number of them. However, remember that every time a data transmission event occurs, the nodes enter into passive and active scan states for periods up to $T_{ADT1.}$ Note that the same T_{ADT1} parameter is notified to all the nodes but the effective time a node spends in active/passive scan is different and is lower for relays several hops away from the source node. Consumption linked to scan mode is not negligible and T_{ADT1} depends on the duration of the T_{listen} period (T_{poll} and N_{PO}) and $TTL_.$ When T_{ADT1} increases (in Fig. 11 we can appreciate T_{poll} values where trend changes occur), the greater the T_{listen} period, the greater the energy consumption.

Logically, both energy saving and delay degrade if the parameter N_{PO}, which is linked to the feasibility and redundancy, is increased. Remember that N_{PO} must be chosen according to the T_{poll} parameter, depending on the BLER (see Table 2). For instance, assuming BLER=10%, for T_{poll}=50 ms, N_{PO}=5 is enough, whereas for T_{poll}=20 ms, N_{PO} =6 is required. In those conditions, with 1 TX/min and TTL=10, the relative consumption is 29.8% with a delay of 2.59 s for T_{poll}=20 ms, whereas consumption is reduced to 18.21% with a delay of 3.8 s for T_{poll}=50 ms. In this case, $T_{poll=}$50 ms seems a good configuration.

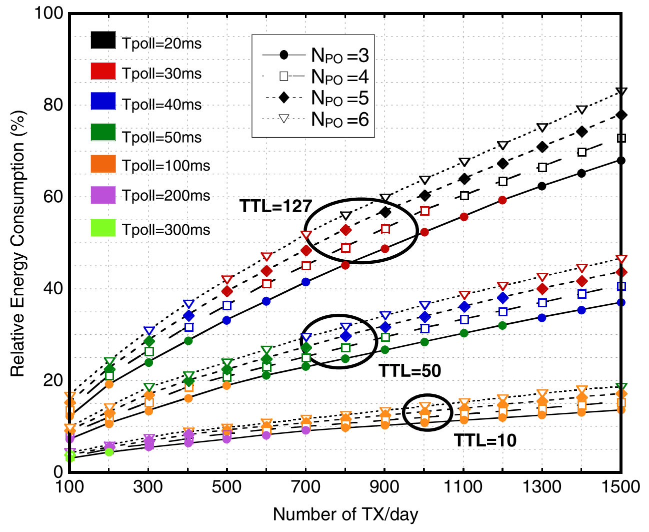


Fig. 13. PSM-DMO proposal. Relative Energy Consumption (%) of PSM-DMO proposal compared with the standard mode implementation as the number of TX/day increases from 100 to 1500 for TTL=10, 50 and 127.

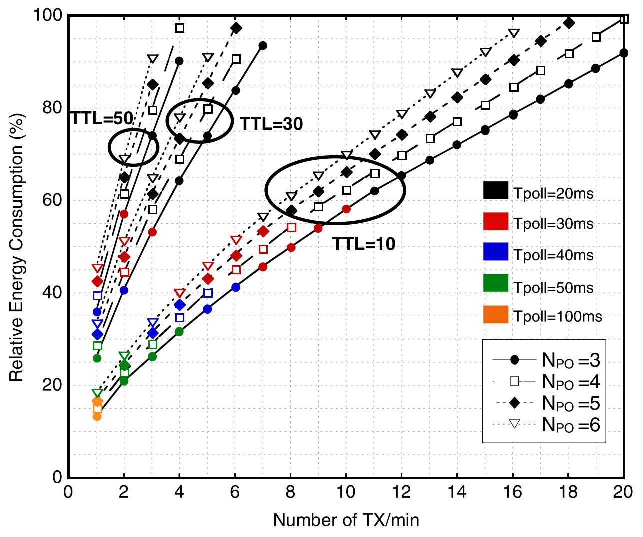


Fig. 14. PSM-DMO proposal. Relative Energy Consumption (%) of PSM-DMO proposal compared with the standard mode implementation as the number of TX/min increases from 1 to 20 for TTL=10, 30 and 50.

However, with 1 TX/min and TTL=50, the relative consumption is 46.6% with a delay of 13.1 s for T_{poll}=20 ms, whereas consumption is reduced only to 44.68% with a delay of 19.1 s for T_{poll}=50 ms. In this case, the option to choose is not clear. However, with TTL=127, the best option from the point of view of both energy saving and delay is T_{poll} =20 ms. In this case, relative consumption is 79% with a delay of 33.22 s.

Note that, even in extreme cases of networks requiring 127 hops and having high traffic, we can still benefit from the mechanism and show energy savings of 21% when BLER=10% and 35% in almost ideal conditions in terms of BLER (BLER=0%). Remember that, in all the cases, parameters have been chosen to compensate collision probabilities.

Fig. 13 shows with more detail how the relative energy consumption grows as long as traffic increases from 100 TX/day to 1500 TX/day (around the 1 TX/min) and for different values of N_{PO} (3–6). In this case, only the minimum relative energy consumption, obtained for the best T_{poll} configuration, is shown. For each T_{poll}, the power consumption grows linearly, being larger the slope as the value of T_{poll} increases.

Certainly, from the trend of the results obtained in Figs. 11–13 we derive that PSM-DMO is an advantageous proposition when network traffic is infrequent. However, the benefits can be extended when the number of hops is low or moderate. Fig. 14 shows the relative energy consumption (in %) for TTL=10, 30 and 50 when traffic grows above 1TX/min. In this case, relative energy consumption is shown for

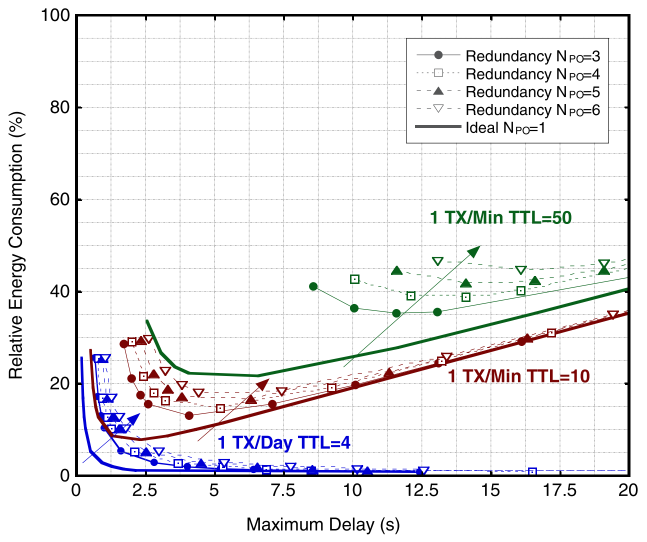


Fig. 15. PSM-DMO proposal. Relative Energy Consumption (%) compared with the standard mode implementation vs Delay.

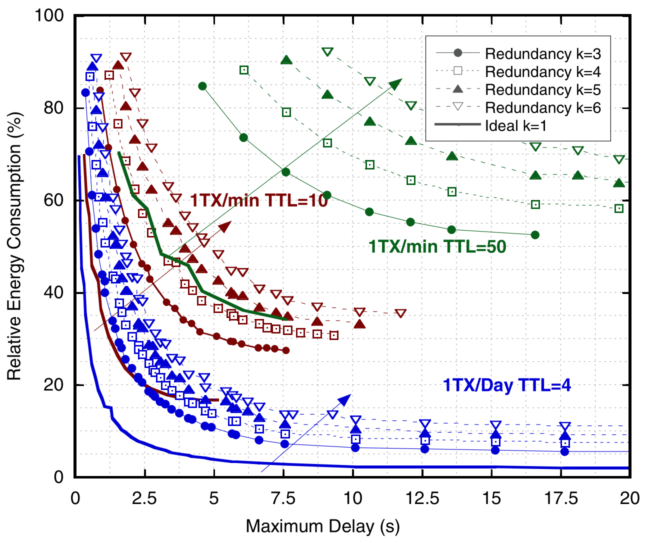


Fig. 16. BMADS proposal. Relative Energy Consumption (%) compared with the standard mode implementation vs Delay.

different values of N_{PO} (3 to 6) and for the T_{poll} value (between 20 and 100 ms) that provides the best performance. When TTL=10, consumption is still reduced around 50% for 8TX/min. When the number of hops increases (TTL=50 in versus TTL=30 or TTL=10), in Fig. 14 we can see again that energy consumption grows faster as traffic increases. Also, as the number of hops increases, the parameter settings that provide the minimum energy consumption correspond with lower T_{poll} values.

Concerning with the number of devices that are sources of data, in general, and not only for PSM-DMO but also standard operation, the performance is affected by an increased number of collisions as the number of transmissions increases. When there is a greater number of devices increasing the number of transmissions, the overall performance gets worse. In any case, PSM-DMO performs better than the standard as long as the overall number of transmissions is infrequent.

In order to compare the PSM-DMO and BMADS [22] proposal, Fig. 15 (for PSM-DMO) and Fig. 16 (for BMADS) quantify the relation between relative energy consumption and delay in three scenarios (1 TX/day and TTL=4 and 1 TX/min for TTL=10 and TTL=50).

In the case of BMADS, as it can be seen in [22], there is not a single optimal configuration for the mechanism but a set of non-dominated solutions to choose from. That is to say, solutions in which the energy consumption cannot be further improved without degrading the maximum delay. Such solutions are Pareto optimal and constitute the configurations to consider and that are represented in Fig. 13. Table 3 summarizes BMADs parameters. To configure the BMADS control sequence that puts the relays into continuous scan, BMADS use N_{TC}^{seq}, N_{TIS}^{seq}, P_{RC}^{seq}, and N_{RIS}^{seq} parameters and calculate the associated values, $netInt^{seq}$ and $pubInt^{seq}$, as defined in [22]. These parameters should be selected in a way that we can create a long enough pattern to ensure that at least one control message (named ADS) is received during the active part of the scanner duty cycle, assuming ideal conditions in terms of collisions or BLER. The scanner duty cycle is controlled by $scanWin^{seq}$ and $scanInt^{seq}$ parameters, which together with the previous ones form the set of parameters to choose from.

To ensure reliability in non-ideal conditions, BMADS introduces the parameter k, which represents the minimum number of consecutive control packets (ADS) to be received to ensure a successful control message reception. That is, to ensure the reception even with up to k-1 consecutive packet with errors. The purpose is similar to that of N_{PO} and repetitions of adv^{seqADT}. Redundancy of the control sequence and the data is controlled with their correspondent P_{RC}^{data}, $N_{TC}^{data}$and R_{RC}^{data}. Although $N_{TC,}$ $N_{TIS,}$ P_{RC}, and N_{RIS} could be different for the data or the BMADS sequence as their target is different, the same level of redundancy is considered for the control messages and data ($N_{TC}^{data}= R_{RC}^{data} = k-1$, $P_{RC}^{data} = 0$). Tuning these parameters, including k, not only affects the reliability of the transmissions but also has a direct impact on the receiver scan cycles. Hence, energy consumption depends heavily on how they are adjusted. As stated above, results in Fig. 16 correspond with the Pareto optimal solutions for k=1 (non-redundancy) and from 3 to 6. Parameter combinations are represented by different points. Unrepresented values of delay are not feasible. Increasing redundancy has a higher impact in large scenarios with higher traffic. This is because continuous scan is kept active for a longer time when the network is larger and such reconfiguration happens with every new transmission. As a result, the effect of network size and traffic volume is multiplicative. Note that even with k=6, important energy savings are still observed with respect to normal Bluetooth Mesh Operation.

Comparing the new proposal (PSM-DMO) with BMADS, under equivalent conditions, we see that PSM-DMO clearly exceeds the performance of BMADS in all cases. For instance, when the number of transmission events is low (1 TX/day) and TTL=4, with a delay up to 2.5 s, relative energy consumption in BMADS is around 20% with k=3, whereas in PSM-DMO is below 3% with N_{PO}= 3, and even lower than 5.5% with N_{PO}= 6 (while is around 38% in BMADS). Thus, in this specific situation, PSM-DMO reduces power consumption 7 times and up to 8 times when higher delays are considerd. As can be seen by comparing Figs. 12 and 15, the gain of PSM-DMO depends on the specific delay threshold (for instance, around 4 times for 1.25 s).

As the traffic (1 TX/min) and the number of hops increases, both schemes tend to the continuous scan mode and therefore the ratio between them is reduced. However, differences between PSM-DMO and BMADS are still very significant. For instance, for TTL=50, N_{PO} =k=3, and 12.5 s, relative energy consumption in PSM-DMO is 35% compared with 55% of BMADS. With N_{PO} =k=6 and 12.5 s, relative energy consumption is 80% in BMADS. However, in this case, in PSM-DMO the minimum delay is 13 s but relative energy consumption drops to 47%. Thus, the gain is significant. We can see that PSM-DMO imposes a higher minimum delay than BMADS when traffic increases and for a high number of hops (*TTL*). However, the energy reduction is significant with a very low impact over the minimum delay.

From the previous discussion, we can conclude that the PSM-DMO proposal provides very significant energy benefits in networks where traffic is infrequent. and, furthermore, when traffic increases if the number of hops is not excessive. In any case, we recall that the proposal is defined as an optional operation mode, eligible when nodes operate in scenarios with applications that are delay tolerant and/or make infrequent transmissions. That is, PSM-DMO needs to coexist with standard mesh operation in other cases.

6. Conclusions

Mesh feature has risen the interest of BLE as an enabler for IoT. However, almost continuous scan requirement stated in the standard and linked to the managed flooding scheme reduces its advantages in terms of energy consumption compared to other alternatives. Particularly, in application scenarios where its backbone is battery-powered and traffic is infrequent. To overcome this limitation, we have proposed a novel strategy, named PSM-DMO, that minimizes the scan periods and thus, significantly reduces the overall power consumption. The proposal, adapted to the BLE mesh specification, allows all nodes of the mesh network to go into scan mode only when there is a node with data to transmit and flooding is required. It replaces the continuous scan by a periodic but asynchronous polling process whereby the nodes interrogate their neighbors about the existence of data to receive or to retransmit through the network. Nodes remain by default in sleep mode between polls in order to save energy and, they only go into scan mode during the period of time the mesh network will be involved in the transmission and dissemination of data. This proposal clearly differs from the concept of Bluetooth Mesh Low Power Node (BM-LPN) and, also, from BMADS proposal previously presented in [22], based on dynamic scan cycles and sending of a new control message sequence that puts the nodes into a continuous scan. The results of the new proposal (PSM-DMO), which minimizes scan periods, show that the approach provides very power-efficient Bluetooth Mesh networks. The evaluation has been made considering the redundancy required by the propagation and interference conditions. It provides very significant energy benefits in networks where traffic is infrequent. For applications that require 1 TX/day, substantial energy savings have been observed, depending on the number of hops. Although there is a compromise between energy saving consumption and delay, over 99.24% and few seconds of delay can be achieved in networks with few hops. Therefore, it is possible to extend a 4400 mAh powered node from less than 40 days to more than 43.7 years. A 98.24% energy saving can be achieved for 127 hops and applications with higher tolerance of delays.

Networks having higher traffic demands (1 TX/min), even when they require a high number of hops, can also benefit from this mechanism. For instance, networks with 50 hops show energy savings above 55% with delays below 12 s. Compared with BMADS proposal, PSM-DMO far outperforms BMADS in all cases, while parameter selection is easier. Energy consumption can be reduced up to 7-8 times compared with BMADS.

The proposal is defined as an optional operation mode, eligible when nodes operate in scenarios with applications that are delay tolerant and/or make infrequent or sporadic transmissions. It does not either require relevant changes in the mesh protocol stack structure although an additional feature is required.

Authorship statement

All persons who meet authorship criteria are listed as authors, and all authors certify that they have participated sufficiently in the work to take public responsibility for the content, including participation in the concept, design, analysis, writing, or revision of the manuscript. Furthermore, each author certifies that this material or similar material has not been and will not be submitted to or published in any other publication before its appearance in the Computer Networks.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work has been supported in part by the Spanish Ministry of Science through the projects RTI2018-099880-B-C32. RTI2018-095684-B-I00 and RTI2018-099063-B-I00 with ERFD funds, and by the Government of Aragon (Reference Group T31_20R).

References

[1] L. Chettri, R. Bera, A Comprehensive Survey on Internet of Things (IoT) Toward 5G Wireless Systems," in, IEEE Internet of Things Journal 7 (1) (2020) 16–32.
[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, Internet of things: a survey on enabling technologies, protocols, and applications, IEEE Commun. Surv. Tutor. 17 (4) (2015) 2347–2376, 10.
[3] S.A. Malek, S.D. Glaser, R.C. Bales, Wireless sensor networks for improved snow water equivalent and runoff estimates, IEEE Access 7 (2019) 18420–18436.
[4] G.A. Akpakwu, B.J. Silva, G.P. Hancke, A.M. Abu-Mahfouz, A Survey on 5G Networks for the Internet of Things: Communication Technologies and Challenges, 6, IEEE Access, 2018, pp. 3619–3647.
[5] C. Perera, C.H. Liu, S. Jayawardena, M. Chen, A Survey on internet of things from industrial market perspective, IEEE Access 2 (1) (2015) 1660–1679.
[6] J. Weiss, R. Yu , “Wireless sensor networking for the industrial IoT,” pp. 1–6, 2015. [Online]. Available: https://www.electronicdesign.com/technologies/iot/article/21800947/wireless-sensor-networking-for-the-industrial-iot.
[7] J.M. Williams, R. Khanna, J.P. Ruiz-Rosero, G. Pisharody, Y. Qian, C.R. Carlson, H. Liu, G. Ramirez-Gonzalez, Weaving the wireless web: toward a low-power, dense wireless sensor network for the industrial IoT, IEEE Microw. Mag. 18 (7) (2017) 40–63, 11.
[8] D. Raposo, A. Rodrigues, S. Sinche, J.S. Silva, F. Boavida, Industrial IoT monitoring: technologies and architecture proposal, Sensors 18 (10) (2018) 3568. http://www.mdpi.com/1424-8220/18/10/3568, 10[Online]. Available:.
[9] S. Jacobs, “Power your wireless sensors for 40 years,” 2013. [Online]. Available: https://www.electronicdesign.com/power-management/article/21795789/power-your-wireless-sensors-for-40-years.
[10] "Saft Batteries for The Internet of Things", Saft Batteries, Levallois-Perret, France, 2019. [Online]. Available: https://bit.ly/3eMbNx8.
[11] A. Kutyła-Olesiuk, M. Zaborowski, P. Prokaryn, P. Ciosek, Monitoring of beer fermentation based on hybrid electronic tongue, Bioelectrochemistry 87 (10) (2012) 104–113.
[12] T. Ojha, S. Misra, N.S. Raghuwanshi, Wireless sensor networks for agriculture: the state-of-the-art in practice and future challenges, Comput. Electron. Agric. 118 (10) (2015) 66–84.
[13] G. Hornero, D. Conde, M. Quílez, S. Domingo, M.P. Rodríguez, B. Romero, O. Casas, A wireless augmentative and alternative communication system for people with speech disabilities, IEEE Access 3 (2015) 1288–1297.
[14] E. Sifuentes, O. Casas, R. Pallas-Areny, Wireless magnetic sensor node for vehicle detection with optical wake-up, IEEE Sens. J. 11 (8) (2011) 1669–1676.
[15] “Mesh profile Bluetooth specification v1.0.1,” Bluetooth SIG, Kirkland, WA USA, 2019. [Online]. Available: https://www.bluetooth.com/specifications/specs/mesh-profile-1-0-1/.
[16] “Bluetooth core specification 5.2,” Bluetooth SIG, Kirkland, WA USA, 2019. [Online]. https://www.bluetooth.com/specifications/specs/core-specification-5-2/.
[17] D. Hortelano, T. Olivares, M.C. Ruiz, Reducing the energy consumption of the friendship mechanism in Bluetooth Mesh, Comput. Netw. 195 (2021), 108172. Volume4 August.
[18] “nRF52840 product specification v1.1.” Nordic Semicond., Trondheim, Norway, 2019, [Online]. Available: https://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.1.pdf.
[19] “nRF52832 product specification v1.4.”, Nordic Semicond., Trondheim, Norway, 2017, [Online]. Available: https://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.4.pdf.
[20] S. Darroudi, R. Caldera-Sánchez, C. Gomez, Bluetooth Mesh energy consumption: a model, Sensors 19 (5) (2019) 1238. https://www.mdpi.com/1424-8220/19/5/1238, 3[Online]. Available:.
[21] “nRF51422 - Bluetooth low Energy, ANT and 2.4 GHz SoC ”, Nordic Semicond., Trondheim, Norway, 2020. [Online]. Available: https://www.nordicsemi.com/products/nrf51422.
[22] D. Perez-Diaz-de-Cerio, J.L. Valenzuela, M. Garcia-Lozano, Á. Hernández-Solana, A. Valdovinos, BMADS: BLE mesh asynchronous dynamic scanning, IEEE Internet Things J. 8 (4) (2021) 2558–2573. Feb.15.
[23] J. Yang, C. Poellabauer, P. Mitra, C. Neubecker, Beyond beaconing: emerging applications and challenges of BLE, Ad Hoc Netw. 97 (2) (2020), 102015.
[24] N. Todtenberg, R. Kraemer, A survey on Bluetooth multi-hop networks, Ad Hoc Networks 93 (10) (2019), 101922.
[25] Y. Murillo, B. Reynders, A. Chiumento, S. Malik, P. Crombez, S. Pollin, Bluetooth now or low energy: should BLE mesh become a flooding or connection oriented network?, in: Proceedings of the IEEE International Symposium on Personal,

Indoor and Mobile Radio Communications, PIMRC Institute of Electrical and Electronics Engineers Inc., 2018, pp. 1–6, 2017-October.
[26] P. Levis, P. Clausen, J. Hui, O. Gnawali, and J. Ko, "The trickle algorithm," Internet Eng. Task Force, RFC 6206, 2011. [Online]. Available: https://tools.ietf.org/html/rfc6206.
[27] M-Way Solutions GmbH, "FruityMesh - the first completely connection-based open source mesh on top of Bluetooth Low Energy (4.1/5.0 or higher)," 2019. [Online]. Available: https://github.com/mwaylabs/fruitymesh.
[28] A.S. Brandao, M.C. Lima, C.J.B. Abbas, L.J.G. Villalba, An energy balanced flooding algorithm for a BLE mesh network, IEEE Access 8 (2020) 97946–97958, 5.
[29] X. Wang, K. Xu, B. Mao, "GreenLink: An Energy Efficient Scatternet Formation for BLE Devices", *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 1836198, 13 pages, 2018.
[30] R.T.E. Dvinge, A. Stalmach, L. Nalpantidis, Connection-Based Bluetooth Mesh Network as a Low Energy Solution for Off-Grid Data Networks, in: Proceedings of the 8th International Conference on Modern Circuits and Systems Technologies, MOCAST 5, Institute of Electrical and Electronics Engineers Inc., 2019, 2019.
[31] A. Beben, A. Bak, M. Sosnowski, Efficient relay node management in BLE mesh networks, Int. J. Electron. Telecommun. 66 (2020). http://www.ijet.pl/index.php/ijet/article/view/10.24425-ijet.2019.130262 [Online]. Available:.
[32] C. Cano, B. Bellalta, A. Sfairopoulou, M. Oliver, Low energy operation in WSNs: a survey of preamble sampling MAC protocols, Comput. Netw. 55 (15) (2011) 3351–3363, 10.
[33] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez, "IPv6 over BLUETOOTH (R) Low Energy (RFC 7668)," 2015. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7668.
[34] B. Luo, Z. Sun, Y. Pang, A. Ahmad, J. Lin, J. Wu, H. Zhang, Neighbor discovery for IPv6 over BLE mesh networks, Appl. Sci. 10 (5) (2020) 1844. https://www.mdpi.com/2076-3417/10/5/1844, 3[Online]. Available:.
[35] "Bluetooth® Core Specification 5.0", Bluetooth SIG, Kirkland, WA USA, 2016. Available: https://www.bluetooth.com/specifications/specs/core-specification-5/.
[36] "Online power profiler", Nordic Semiconductor, Trondheim, Norway, 2020. [Online]. Available: https://devzone.nordicsemi.com/power/w/opp.
[37] Á. Hernández-Solana, D. Perez-Diaz-de-Cerio, A. Valdovinos, J.L. Valenzuela, Proposal and evaluation of BLE discovery process based on new features of bluetooth 5.0", Sensors (Switzerland) 17 (9) (2017) 1988.
[38] D. Perez Diaz de Cerio, A. Hernandez-Solana, A. Valdovinos, J.L. Valenzuela, Low-cost test measurement setup for real IoT BLE sensor device characterization, Measurement 135 (2019) 814–827.
[39] "Bluetooth® Core Specification 4.2", Bluetooth SIG, Kirkland, WA USA, 2014. Available: https://www.bluetooth.com/specifications/specs/core-specification-4-2/.
[40] Á. Hernández-Solana, D. Pérez-Díaz-De-Cerio, A. Valdovinos, J.L. Valenzuela, Anti-collision adaptations of BLE active scanning for dense IoT tracking applications, IEEE Access 6 (2018) 53620–53637.
[41] B. Al Nahas, S. Duquennoy, O. Landsiedel, Concurrent Transmissions for Multi-Hop Bluetooth 5, in: Proceedings of the International Conference on Embedded Wireless Systems and Networks, EWSN, ACM Digital Library, 2019, pp. 130–141.
[42] CC2650MODA SimpleLinkTM Datasheet, Texas Instrum., Dallas, TX, USA, 2018. [Online]. Available: http://www.ti.com/lit/ds/symlink/cc2650moda.pdf.
[43] D. Perez Diaz de Cerio, A. Hernandez, J.L. Valenzuela, A. Valdovinos, Analytical and experimental performance evaluation of BLE neighbor discovery process including non-idealities of real chipsets, Sensors (Switzerland) (2017).

Ángela Hernández-Solana obtained the Engineer of Telecommunications and Ph.D. degrees from the Universitat Politècnica de Catalunya (UPC), Spain, in 1997 and 2005, respectively. She has been working at UPC and at UNIZAR, where she is an Associate Professor since 2010. She is member of the Aragón Institute of Engineering Research (I3A). Her research interests include 5G/4G technologies, heterogeneous communication networks and mission-critical communication networks, with emphasis on transmission techniques, radio resource management and quality of service, mobility management and planning and dimensioning of mobile networks.

David Pérez-Díaz-de-Cerio obtained the MSc in telecommunications Enginnering degree in 2003. That year he joined the WiComTec research group of the Signal Theory and Communications department as a collaborating lecturer. All his lectures are held at the Escola d'Enginyeria de Telecomunicació i Aeroespacial de Catalunya for Bachelor's degree, second-cycle and MAST master students. In 2010, he received the PhD degree in Telecommunication from the Technical University of Catalonia (UPC). He has participated as a consultant in several local and European projects in addition to other public and private funded projects. His research interests are wireless communications systems; especially those based on the IEEE 802.X standards and their use in e-health applications and IoT.

Mario Garcia-Lozano is a Ph.D. in Telecommunications Engineering from the Universitat Politecnica de Catalunya (UPC, Barcelona-TECH) since 2009. Dr. Garcia-Lozano has >20 years of experience in different radio network planning and optimization activities both at the academia and industry. He is currently an associate professor at UPC and his research activities are focused on the field of radio resource management and the optimization of wireless networks. He has actively participated in >25 competitive research projects and contracts with the industry. He is recipient of 3 best paper awards and was the advisor of the student team that won the international competition for mobile network planning organized by the company ATDI.

Antonio Valdovinos Bardají obtained the Engineer of Telecommunications and Ph.D. degrees from the Universitat Politècnica de Catalunya (UPC), Spain, in 1990 and 1994, respectively. He has been working at UPC and at the University of Zaragoza (UZ), where he is a Full Professor since 2003. His research interests include 5G/4G technologies, heterogeneous communication networks and mission-critical communication networks, with emphasis on transmission techniques, radio resource management and quality of service, mobility management and planning and dimensioning of mobile networks.

Jose Luis Valenzuela obtained the Engineer of Telecommunication and Ph.D. degrees from the Universitat Politecnica de Catalunya (UPC), Barcelona, Spain, in 1993 and 1997, respectively. He is currently an Associate Professor in the Signal Theory and Communications Department, UPC. After graduation he was concerned with equalization techniques for digital systems. He has also been working on the field of digital communications with particular emphasis on digital radio and its performance under multipath propagation conditions. His research interest is in the field of wireless sensor networks and wireless communications systems.