IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 24, 2020, accepted August 15, 2020, date of publication September 7, 2020, date of current version September 23, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3022434

Wireless Sensor Networks and TSCH: A Compromise Between Reliability, Power Consumption, and Latency

STEFANO SCANZIO[1], (Member, IEEE), MOHAMMAD GHAZI VAKILI[2], GIANLUCA CENA[1], (Senior Member, IEEE), CLAUDIO GIOVANNI DEMARTINI[2], (Senior Member, IEEE), BARTOLOMEO MONTRUCCHIO[2], (Member, IEEE), ADRIANO VALENZANO[1], (Senior Member, IEEE), AND CLAUDIO ZUNINO[1], (Member, IEEE)
[1]National Research Council of Italy (CNR-IEIIT), 10129 Turin, Italy
[2]Dipartimento di Automatica e Informatica, Politecnico di Torino, 10129 Turin, Italy

Corresponding author: Mohammad Ghazi Vakili (mohammad.ghazivakili@polito.it)

ABSTRACT Reliability, power consumption, and latency are the three main performance indicators of wireless sensor networks. Time slotted channel hopping (TSCH) is a promising technique introduced in the IEEE 802.15.4 standard that performs some steps ahead in the direction of the final dream to meet all the previous requirements at the same time. In this article, a simple and effective mathematical model is presented for TSCH that, starting from measurements performed on a real testbed, permits to characterize both the network and the surrounding environment. To better characterize power consumption, an experimental measurement campaign was purposely performed on OpenMote B devices. The model, which was checked against a real 6TiSCH implementation, can be employed to predict network behaviour when configuration parameters are varied, in such a way to satisfy different application contexts. Results show that, when one of the three above indices is privileged, unavoidably there is a worsening of the others.

INDEX TERMS Energy consumption, Internet of Things, low latency, power consumption, reliability, wireless sensor networks, time slotted channel hopping (TSCH).

I. INTRODUCTION

Wireless sensor networks (WSN) [1] are becoming more and more one of the core enabling technologies for the Internet of Things (IoT). They permit simple devices, often powered on batteries and typically equipped with sensors (and, sometimes, actuators), to be easily interconnected and integrated in a monitoring/control system. When actuation is possible, they are typically referred to as wireless sensor and actuator networks (WSAN).

WSNs, and in general WSANs, are gaining ground in many application contexts characterized by very different and often conflicting requirements. Notable examples are environmental monitoring [2], smart and precision agriculture [3], greenhouse automation [4], heating, ventilation, and air conditioning (HVAC) systems [5], natural disaster management [6], wireless body area networks (WBAN) for healthcare [7], military applications [8], unmanned surveillance [9], and retrofitting of large industrial plants for monitoring and diagnostics [10].

Depending on the specific field of application, a number of requirements are demanded to the WSN. In most cases, reduced power consumption is the primary goal [11]–[15]. Typically, WSN nodes (also referred to as motes) are battery-powered, and in some cases they are deployed in harsh areas or in places that are difficult to reach (e.g., when used for environmental monitoring and natural disaster management), making battery replacement a cumbersome task.

As a matter of fact, WSNs cannot be used for time-critical applications requiring extremely low latencies and jitters (in the order of a few milliseconds), such as cable replacement [16], [17]. In these cases, specific high-performance wireless solutions have been recently defined, often identified with the term ultra-reliable and low-latency communications

The associate editor coordinating the review of this manuscript and approving it for publication was Yu Liu.

(URLLC) [18]. Nevertheless, many contexts in the real world may benefit from (and sometimes demand for) the ability of the underlying network to ensure relatively short (seconds) and bounded latency on packet delivery. This is the case, e.g., of industrial monitoring and surveillance applications.

Similarly to power consumption and transmission latency, also the required degree of reliability the network must ensure depends on the specific application. In theory, this goal can be easily achieved by means of Automatic Repeat reQuest (ARQ) techniques [19], which rely on confirmed message exchanges and operate by retransmitting those frames for which no corresponding acknowledgement (ACK) was received. In practice, retransmission increases both power consumption and latency.

Taking all above requirements into account at the same time is typically a complex task to deal with. To this aim, a popular solution is time slotted channel hopping (TSCH) [20]. TSCH was first proposed as an enhancement of IEEE 802.15.4 (known as 802.15.4e) and then included into the most recent version of the related standard specification [21].

Currently, many standards for wireless networks exist [22] that can be profitably adopted in practical implementations of the Internet of Things (IoT) paradigm, and often several heterogeneous communication technologies coexist within the same system, which are transparently exploited by applications [23]. Popular IoT solutions that rely on IEEE 802.15.4 for frame exchanges include, e.g., ZigBee and WIA-PA. Concerning IEEE 802.15.4e, two operating modes are defined, i.e, the deterministic and synchronous multi-channel extension (DSME) and TSCH (used by WirelessHART, ISA 100.11a, and 6TiSCH). Other technologies commonly adopted in IoT are IEEE 802.11 (Wi-Fi), 4G/5G, LoRa, and Bluetooth Low Energy (BLE).

The analysis presented in this article specifically refers to TSCH, and experimental results were obtained on motes complying to the IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) protocol [24]–[26]. We opted for TSCH because it shows a deterministic behavior [27] for what concerns the metrics for IoT systems we took into account, namely *reliability*, *power consumption*, and *latency*. Protocol behavior can be finely tuned by means of a number of parameters, most of which can be easily configured by the user. However, since the above metrics are deeply intertwined, it is generally impossible to optimize one without worsening the others. Therefore, proper network configuration to meet the application requirements must necessarily rely on holistic approaches.

The main contribution of this article is a simple and effective mathematical formulation that describes the behavior of a TSCH network, with the ultimate goal to ease tuning procedures. The model depends on both protocol configuration parameters and estimated quantities aimed at characterizing the network and the surrounding environment, obtained from experimental measurements performed on a real setup. Logged data were also used for validating the model.

The second relevant contribution of the paper is the experimental characterization of the power consumption of OpenMote B devices. Although these motes are quite widespread, as they are currently employed for developing and testing any newly added feature or improvement of both 6TiSCH and the OpenWSN operating system, to the best of our knowledge such a kind of analysis is not available in the literature yet. From our point of view, this characterization is required to enable an adequate and truthful estimation of the power consumption in the network model.

In the following, Section II describes the main aspects of TSCH. The network and power consumption models are presented in Section III and Section IV, respectively. Details about the experimental analysis are provided in Section V, while considerations on some relevant, practical application contexts with respect to performance indicators are reported in Section VI. Finally, some concluding remarks are given in Section VII.

II. TSCH

The TSCH mode of the IEEE 802.15.4-2015 [21] standard was first defined in 2012, and was initially included in the IEEE 802.15.4e amendment [28]. On the one hand, TSCH is an effective way to counteract disturbance caused by either electromagnetic noise or transmissions on the wireless medium performed by other communication technologies that operate in the same (or on an overlapping) frequency range. On the other hand, it increases communication predictability and reduces power consumption.

More in detail, time in a TSCH network is divided into slots of fixed duration, organized in slotframes made up of a defined number N_{slot} of slots. Slotframes repeat in time and are exploited by a *time slotted* mechanism for controlling wireless medium access. This requires all motes to be time-synchronized [29], [30]. In commonly available implementations, the duration T_{slot} of a slot ranges from 10 ms to 20 ms.

A second mechanism is additionally defined, known as *channel hopping*, aimed at constantly changing the transmission channel according to a known pattern. In our implementation, the 16 channels defined by IEEE 802.15.4 in the 2.4 GHz ISM band are exploited, each of which has a width of 5 MHz, lying in the range between 2.405 GHz (channel 11) and 2.480 GHz (channel 26).

From the motes' point of view, the wireless spectrum is conceptually partitioned in frequency and time through one (or more) matrices, where the rows of the matrix represent the 16 channels while each column represents a specific slot in the slotframe [31]. An example of this matrix is shown in Fig. 1. According to the TSCH terminology, rows identify *channel offsets*, columns refer to *slot offsets*, and the matrix defines the *slotframe* usage. In particular, a *cell* located at the intersection between a specific row and a specific column of the matrix (i.e., channel and slot offsets) describes a *link* between neighbor nodes at the data-link layer.

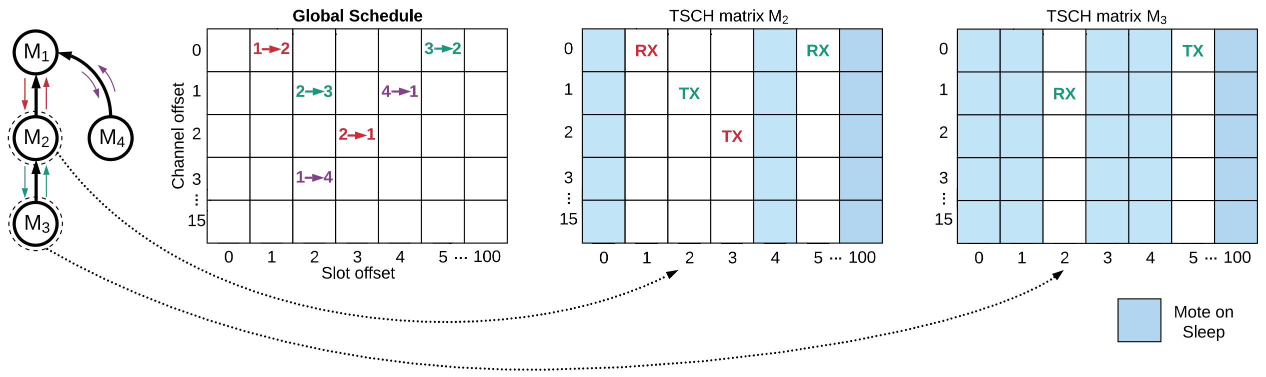


FIGURE 1. Example of a TSCH matrix defining the slotframe usage: global schedule (on the left) and local, trimmed-down copies (on the right).

Each slot in a TSCH network is uniquely identified by the absolute slot number (ASN), which is a counter initialized to 0 at the beginning of the network formation and incremented by one on every slot. A mechanism is defined that enables all nodes to know, at any time, the global value of ASN for the network they belong to. This permits each node to determine the offset of the current slot in the slotframe by simply computing ASN mod N_{slot}. The physical channel used for transmission in any given cell is selected according to a pseudo-random function that is known by all the nodes of the network and depends on both the ASN value and the related channel offset. Typically, the transmission channel changes in each repetition of the slotframe.

On the one side, cells within the matrix can be configured to obtain the desired schedule [32], [33], which, thanks to the time-slotted mechanism, increases network determinism by preventing intra-network collisions. For instance, a sequence of cells that defines a multi-hop path between motes M_1 and M_3 is shown in Fig. 1. On the other side, the channel hopping mechanism is effective to counteract narrowband disturbance and external interference that may affect communication. In particular, every time the transmission in a cell is corrupted, the physical channels used for retries in the following slotframes will be very likely different.

Concerning a single mote, generally speaking a scheduled cell in the matrix can be configured for either transmission (TX) or reception (RX). When no cell is scheduled for the current time slot, the mote can switch off its radio module (transceiver), consequently saving energy. If the width of the matrix (i.e., N_{slot}) is large and the number of RX cells is small, the mote is enabled to spend most of the time in a deep-sleep state. This means reduced duty cycles and permits noticeable energy saving. Regarding TX cells, in optimized implementations motes wake up only if packets ready to be sent are pending in transmission buffers, whose destination address matches the link described by the current cell.

A cell is defined as *shared* in the case more than one mote is enabled to transmit in it. The resulting behaviour resembles slotted-Aloha, and a mechanism based on random backoff is foreseen to deal with possible collisions. Typically, shared cells are used for frames devoted to network management, while for application data exchanges (which is what we want to model in this article) dedicated (non-shared) cells are more appropriate (and typically exploited).

It is worth remarking that TSCH is part of the medium access control (MAC), and it manages only the communication schedule between neighbour nodes at the data-link layer. Matrices that define slotframe and spectrum usage are configured by means of other protocols operating at higher levels of the protocol stack.

Many papers in the scientific literature focus on techniques for configuring the TSCH matrix [32], [34]–[37], but only a few propose mathematical models to describe performance in the presence of transmission errors. In particular, some theoretical analyses about how a Wi-Fi interfering traffic may affect legacy IEEE 802.15.4 networks (i.e., not based on TSCH or DSME) were reported in [38], [39]. Instead, a model based on Markov chains that describes the traffic on shared cells was proposed in [20], [40]. Finally, some preliminary models describing communication over dedicated cells in TSCH were sketched in [19] and then refined in [41], but without taking into account energy consumption and multi-hop networks.

As said before, experimental evaluation in this article was performed on devices complying to the 6TiSCH protocol. This is for two main reasons. Firstly, 6TiSCH is the most recent among the solutions based on TSCH for which devices are available off-the-shelf (even though their firmware has still to be finalized). A second point in favour of 6TiSCH is that it adopts IPv6 at the network layer (according to the outcomes of the 6LoWPAN project), which fully adheres to the IoT paradigm by permitting each mote to be assigned a unique address within the whole Internet scope. Concerning higher-layer protocols in the 6TiSCH suite, the most relevant ones are the 6TiSCH Operation Sublayer Protocol (6P) [42] to configure TSCH matrices and the Routing Protocol for Low-Power and Lossy Networks (RPL) [43] to manage multi-hop communication between pairs of nodes by creating and maintaining a tree network topology.

A taxonomy of the scientific works about WSN/WSAN referred to in this article, categorized according to the specific aspects they investigate, is reported in Table 1.

TABLE 1. Taxonomy of the scientific works cited in the paper.

Topic	References
WSN/WSAN applications	[2]–[10]
TSCH/6TiSCH	[20], [24]–[27], [31]
IEEE 802.15.4 specifications	[21], [28]
Scheduling and TSCH configuration	[32]–[37]
Mathematical models	[19], [20], [38]–[41]
WSN/WSAN high-level protocols	[42] (6P), [43] (RPL), [46] (CoAP)
Power consumption	[11]–[15], [48], [53]

III. MATHEMATICAL MODEL

Star topologies, in which every node is one hop away from the root, are typically found in many wireless networks conceived for industrial systems (see, e.g., [44]). An example of this kind of architectures can be found in robotized production cells, where sensors fastened to moving (or mobile) parts of the equipment have to be wirelessly connected to a centralized control unit. In this case, all nodes but the root are leaves and conceptually lie at the same network level.

In the case not all nodes of the network are in line of sight, a proper routing mechanism is required for communication, which relies on packet relays performed by intermediate nodes according to a multi-hop transmission technique. A tangible advantage of these solutions is the ability to cover larger areas, at the price of an increased communication latency. Multi-hop WSNs often rely on a multi-level tree topology that stems from a root node [45]. As a matter of fact, mesh topologies also exist where routes between nodes are (somehow) arbitrary, but they are less common in practice and will not be analyzed in this work. Many WSN protocol stacks, like 6TiSCH, rely on RPL to obtain a tree logical topology out of a mesh physical topology.

The mathematical model proposed in this section describes the most general case of a multi-level tree topology, but it can be easily adapted to single-level star topologies. N_{level} is the distance (in terms of levels) between the farthest leaf mote and the coordinator of the WSN (i.e., the *root* of the tree). Clearly, in the single-level case, $N_{level} = 1$. In this analysis, our attention is focused on the request/response paradigm, which is employed, e.g., by the Constrained Application Protocol (CoAP) [46], a quite relevant solution in modern WSNs. Hence, every interaction between two motes consists in a request packet that traverses the path in the forward direction, immediately followed by a response packet in the reverse direction. In the following, with a slight abuse of notation, and in order to simplify equations, we will use the term ‘‘packet i’’ to refer to both the request packet and the related response packet. This means that two-way packet exchanges in star topologies can be described by setting $N_{hop} = 2$ (one hop from the root to the target mote and another from that mote back to the root). Likewise, the number of hops needed to query a node at level l is $N_{hop} = 2 \cdot l$. When a leaf node is queried in a balanced tree, $N_{hop} = 2 \cdot N_{level}$.

The following analysis has been split in two main phases. In the first phase, reported in Sections III-A, III-B, and III-C, we describe the network model and how it can be fit to a real setup. In particular, we show that estimates of the performance indicators we are interested in (reliability, power consumption, and latency) can be evaluated from the model, and that many parameters of the model can be derived directly from a set of measurements performed on a simple experimental setup. In the second phase, described in Section III-D, the model parameters derived directly from the setup are used to indirectly obtain additional derived quantities, which in turn permit to model to a complete extent the expected behavior of a TSCH network when its protocol parameters are varied. The proposed methodology permits to relate model/protocol parameters and performance indicators, and enables a swift network configuration starting from the requirements of applications.

When model parameters are estimated in the first phase, queuing insides nodes is not considered. In other words we assumed that, for any given mote, only one packet with the same destination for the next hop can be found in the queue at any given time. From our viewpoint, this condition is not very restrictive. In fact, it can be easily met if the packet rate with which measurements are performed is much below the available capacity of the path [47]. It is worth pointing out that, in most real application scenarios, the period of data transmissions on a WSN is quite long, which means that queuing phenomena are typically negligible. In the following analysis, we considered paths with a single dedicated cell for each hop, i.e., the matrix of each mote does not contain more than one cell targeted to the same destination. Again, this can be considered a typical configuration for such matrix.

The main quantities used in the model, including protocol parameters and performance indicators, are summarized in Table 2.

A. RELIABILITY

For applications demanding high *reliability*, it is of utmost importance that all transmitted packets (or, at least, the vast majority of them) are correctly delivered to destination. ARQ schemes, like the one exploited in TSCH, permit the transmission of every packet to be reiterated (upon failures) on each traversed link up to the retry limit. The likelihood for any packet to eventually reach the target node is related to the *frame error probability* ϵ (i.e., the probability for a single attempt in the frame transmission process between neighbor nodes to fail), the maximum number N_{tries} of allowed tries per frame at the MAC layer (retry limit plus one), and the number N_{hop} of hops performed at the routing layer (we considered both directions). As a consequence, the *packet delivery probability* on the whole (two-way) path between

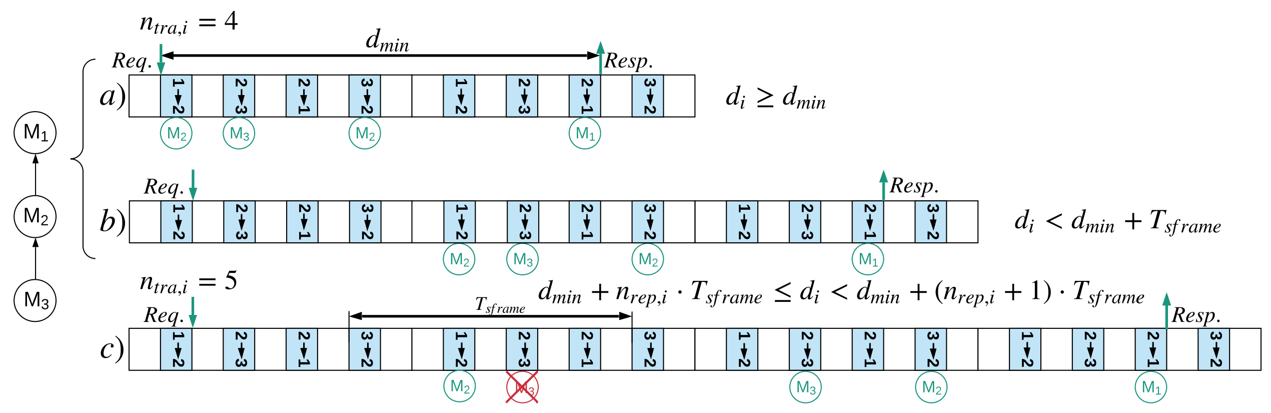

$n_{tra,i} = 4$
d_{min}
Req.
Resp.
a)
$d_i \geq d_{min}$
b)
$d_i < d_{min} + T_{sframe}$
$n_{tra,i} = 5$
c)
T_{sframe}
$d_{min} + n_{rep,i} \cdot T_{sframe} \leq d_i < d_{min} + (n_{rep,i}+1) \cdot T_{sframe}$
M1
M2
M3
1→2
2→3
2→1
3→2

FIGURE 2. Example of request/response iteration in TSCH without (cases a and b) and with (case c) transmission errors ($n_{rep,i} = n_{tra,i} - N_{hop}$ represents the overall number of retransmissions performed for the i-th packet on the two-way path).

TABLE 2. Glossary of quantities.

Quantity	Description
N_{slot}	Number of slots in the slotframe
T_{slot}	Duration of a slot
T_{sframe}	Duration of the slotframe
N_{tries}	Maximum number of allowed attempts per frame
N_{level}	Maximum distance between a mote and the root
N_{hop}	Number of hops between source and destination motes
N_{sam}	Number of samples collected in the experiment
T_{app}	Generation period of packets at the application level
$N_{tra,i}$	Actual number of frames transmitted to deliver packet i
N_{lost}	Measured number of lost packets
$\hat{N}_{lost}$	Estimated number of lost packets
$\hat{n}_{tra}$	Average number of frames transmitted per delivered packet
P_{lost}	Packet loss ratio
ϵ	Frame error probability
ϵ_{pkt}	Packet loss probability
$p_{\rm nr}$	Probability that no retries are made for a packet
E	Total energy consumption
E^f_{tx}	Energy consumption to transmit a confirmed frame
E^f_{rx}	Energy consumption to receive a confirmed frame
E^f_{listen}	Energy consumption for idle listening (single slot)
$\hat{N}_{tra}$	Total number of frames transmitted in the experiment
f_{tra}	Rate of frame exchanges (including retries)
f_{listen}	Rate at which idle listening occurs
d_i	Latency experienced by packet i
d_{min}	Minimum latency
μ_d	Mean latency
σ_d	Standard deviation of latency
d_{p99}	99-percentile of latency
d_{max}	Maximum latency
Max_d	Theoretical worst-case latency

The hat symbol (ˆ) over a quantity denotes estimated values

requester and responder, also termed *reliability*, equals one minus the *packet loss probability* ϵ_{pkt} and can be obtained as

$$1 - \epsilon_{pkt} = \left(1 - \epsilon^{N_{tries}}\right)^{N_{hop}}. \tag{1}$$

Above equation holds if the three following assumptions are verified. First, the value of ϵ for the transmission between any two given motes must be the same irrespective of the direction (upward or downward). This reasonably holds, provided that the radio modules of motes are similar. Second, the value of ϵ must not vary over time. While this is generally untrue, experiments performed on a 6TiSCH network deployed in a real environment and subject to non-negligible interference caused by several nearby Wi-Fi networks [41] showed that this approximation is often acceptable. Third and last, the value of ϵ has to be the same for all the links of the path. This is mostly true for small networks based on a star topology but, when $N_{level} \geq 2$, it may not be verified in some circumstances, because the amount of disturbance and interference could differ sensibly over the coverage area of the WSN (which for multi-hop configurations can be quite large). Not unreasonably, we can assume that the spatial distribution of interfering Wi-Fi devices (and, in general, of equipment exploiting wireless communication technologies) is more or less homogeneous, and the same holds for the traffic they generate. In fact, placement of access points is often planned by IT managers with this goal in mind.

The frame error probability ϵ can be estimated starting from a set of measurements concerning round trip times. Let d_i be the network latency, as seen by the originator, between the transmission time of the packet conveying the i-th request and the reception time of the packet that bears the related response (coming from the target mote). The quantity d_i is defined only for successful transactions, where both the request and the response packets are correctly delivered.

The overall number $n_{tra,i}$ of transmission attempts actually performed on air for the packets involved in any given request/response iteration i, all hops considered (in both directions), can be inferred from the duration d_i using the following equation

$$n_{tra,i} = \left\lfloor \frac{d_i - d_{min}}{T_{sframe}} \right\rfloor + N_{hop}, \tag{2}$$

where d_{min} is the minimum communication latency, evaluated over all the experiments (that depends on the configuration of the cells in the TSCH matrix), and $T_{sframe} = N_{slot} \cdot T_{slot}$ represents the duration of the TSCH slotframe.

To ease understanding of (2), an example is shown in Fig. 2 about a possible configuration of the TSCH matrix for a network including 3 motes. Mote M_1 is the root, M_2 its child, and M_3 (not in line of sight with M_1) the child of M_2. The notation $x \rightarrow y$ in a cell of the TSCH matrix denotes a cell reserved for transmission from M_x to M_y. For instance, $1 \rightarrow 2$ specifies that the cell is reserved for transmission from M_1 to M_2, while $2 \rightarrow 1$ means it is reserved for transmission in the opposite direction, i.e., from M_2 to M_1.

In the case of a request/response iteration between the root M_1 and the leaf mote M_3 performed without transmission errors (i.e., when no retransmissions take place), $n_{tra,i} = 4$ and latency is in the range $d_{min} \leq d_i < d_{min} + T_{sframe}$. In particular, when d_i equals (or is slightly above) d_{min} (Fig. 2.a), this means that the packet to be sent was queued on M_1 just before cell $1 \rightarrow 2$. Since packet generation by applications is not synchronous with the TSCH matrix, the time when the transmission request is issued relative to the slotframe boundaries can be modeled as a random variable uniformly distributed between 0 and T_{sframe}. The maximum delay in the case of no retransmissions is bounded by $d_{min}+T_{sframe}$, and is experienced when the packet is queued immediately after the beginning of the $1 \rightarrow 2$ cell, which means that it is sent in the next $1 \rightarrow 2$ cell (Fig. 2.b). Finally, when transmission errors are experienced, the latency d_i is increased by the duration of one slotframe (T_{sframe}) per every frame retransmission. For instance, in the case a single retransmission is performed (Fig. 2.c), on any link, bounds $d_{min}+T_{sframe} \leq d_i < d_{min}+2\cdot T_{sframe}$ can be defined on latency. Equation (2), which permits to calculate the actual number of transmission attempts on air, is derived directly from these bounds on latency.

The probability p_{nr} that a packet is correctly exchanged with the minimum number of transmission attempts (i.e., the probability that no retransmissions are performed for it) can be estimated from experiments as

$$p_{\mathrm{nr}} = (1-\epsilon)^{N_{hop}} = \frac{\left|\left\{i \mid n_{tra,i} = N_{hop}\right\}\right|}{N_{sam}}, \tag{3}$$

where $\left|\left\{i \mid n_{tra,i} = N_{hop}\right\}\right|$ is the number of packets that reached the destination after exactly N_{hop} transmission attempts (on the whole path), while N_{sam} is the number of samples collected in the experiment, i.e., the number of request/response iterations executed by the application.

From (3), the frame error probability can be easily obtained as

$$\epsilon = 1 - \left(\frac{\left|\left\{i \mid n_{tra,i} = N_{hop}\right\}\right|}{N_{sam}}\right)^{\frac{1}{N_{hop}}}. \tag{4}$$

Actually, this estimated value for ϵ takes into account the error rates on each one of the 16 different channels used for transmission by TSCH. In the case of a multi-level network, it also ‘‘aggregates’’ the failure probabilities on all the links that make up the path between the source and the destination of the request/response exchange, and provides a simple yet effective model of the overall behaviour of the wireless spectrum in the place where the WSN is deployed. As said before, this approximation is only acceptable if the error rates on the links composing the path do not differ significantly.

B. POWER CONSUMPTION

Analysis of *power consumption* is rather important in WSNs, since motes cannot usually count on a stable external power source but must rely on batteries and/or energy harvesting. A simple but quite effective power consumption model [48] subdivides the total energy consumption into separate contributions

$$E = E_{tx} + E_{rx} + E_{listen} + E_{cpu} + E_{sleep}, \tag{5}$$

where E_{tx} is the overall energy required for transmission (that includes transmission of data frames and reception of the related ACK frames), E_{rx} is the overall energy required for frame reception (that includes reception of data frames and transmission of the related ACK frames), and E_{listen} is the energy consumed to listen to the channel when waiting for the reception of frames and the related slots remain idle (*idle listening*).

In this work, the two contributions E_{cpu} and E_{sleep} have been embedded in the same quantity $E_{comp} = E_{cpu} + E_{sleep}$, which represents the energy consumed by the mote when no specific application runs on it that uses the network. In the following, we concentrate only on the energy consumed for communication. However, the measured value of E_{comp} for the motes we used in the experimental campaign is provided in Section IV.

The energy consumption E_{net} of the network component over a given time interval can be directly derived from the number n_{listen} of cells reserved for transmission (as per the TSCH matrix) that remain unused, in which receiving nodes uselessly sense the channel, and the number n_{tra} of cells in which transmission is actually performed

$$\begin{aligned} E_{net} &= E_{tx} + E_{rx} + E_{listen} \\ &= n_{tra} \cdot \left(E^f_{tx} + E^f_{rx}\right) + n_{listen} \cdot E^f_{listen}, \end{aligned} \tag{6}$$

where E^f_{tx} and E^f_{rx} correspond to the energy consumed to send and receive a single frame (data plus the related ACK), respectively, while E^f_{listen} is the energy consumed to listen to the channel when a reserved cell is not used.

Equation (6) can be rewritten in terms of the power P as

$$P = f_{tra} \cdot \left(E^f_{tx} + E^f_{rx}\right) + f_{listen} \cdot E^f_{listen}, \tag{7}$$

where f_{tra} is the mean number of confirmed frame transmission attempts per second actually performed on air by all motes (i.e., the mean overall frame transmission rate), while f_{listen} represents the mean number of cells allocated for reception over a time span of one second in which idle listening occurred.

The rate f_{tra} can be obtained dividing the total number N_{tra} of frames transmitted on air in the experiment by the overall duration of the experiment itself

$$f_{tra} = \frac{N_{tra}}{T_{app} \cdot N_{sam}}, \tag{8}$$

where T_{app} is the period with which requests are repeatedly issued by the originating mote.

Instead, the rate f_{listen} was obtained by subtracting N_{tra} (i.e., the number of cells that were actually used in the whole experiment) from the total number of cells reserved for transmission in the same interval, and then dividing by the duration of the experiment

$$f_{listen} = \frac{1}{T_{app} \cdot N_{sam}} \cdot \left(\frac{N_{hop} \cdot T_{app} \cdot N_{sam}}{T_{sframe}} - N_{tra} \right), \quad (9)$$

which can be rewritten as the slot repetition frequency $1/T_{slot}$ multiplied by the fraction of slots in the slotframe reserved for the transmission of the considered request/response packets, minus the mean overall frame transmission rate f_{tra}

$$f_{listen} = \frac{N_{hop}}{T_{sframe}} - f_{tra} = \frac{N_{hop}}{T_{slot} \cdot N_{slot}} - f_{tra}. \quad (10)$$

Quantity N_{tra} is composed of two contributions: the number N_{tra}^{deliv} of transmitted frames associated to packets correctly delivered to destination and the number N_{tra}^{lost} of transmitted frames associated to packets that went lost

$$N_{tra} = N_{tra}^{deliv} + N_{tra}^{lost}. \quad (11)$$

N_{tra}^{deliv} can be computed directly from the $n_{tra,i}$ values inferred by applying (2) to experimental samples

$$N_{tra}^{deliv} = \sum_{i=1}^{N_{sam}} n_{tra,i}. \quad (12)$$

Instead, the value of N_{tra}^{lost} cannot be evaluated directly in our experimental setup. In fact, latency is not defined for lost packets, which implies that (2) cannot be applied. An estimate of it, we denoted $\hat{N}_{tra}^{lost}$, can be derived from the value of ϵ provided by (4) by means of (13), as shown at the bottom of the page, which will be introduced and explained in Section III-D. As a consequence, in the following analysis, in the place of N_{tra} we will actually use its estimate $\hat{N}_{tra}$, obtained by substituting $\hat{N}_{tra}^{lost}$ in (11). It is worth remarking that, in the case no packets are lost in the experiment (i.e., when $N_{lost} = 0$), the quantity N_{tra} can be derived directly from experimental data, and the same holds for f_{tra} and f_{listen}.

Equations (8) and (10) hold only when the generated traffic does not exceed the allocated network capacity, that is, if $f_{tra} \leq N_{hop}/T_{sframe}$. Otherwise, network behavior is unstable and the number of packets queued in the motes' transmission buffers would grow indefinitely. Instead, equation (2), which uses the round-trip time d_i measured on the given path to compute the number $n_{tra,i}$ of transmission attempts performed for the i-th request, holds only if packets never experience queuing delays due to previously buffered packets that use the same outgoing cell. Clearly, the same limitation also affects all the formulas that depend on (2). However, as highlighted in Section III-D, the approximation above is acceptable provided that the traffic injected in the network is low. Since disturbance and interference on air do not depend in any way on queuing, ϵ can be reliably estimated by setting T_{app} for measurements long enough, so as to prevent buffer overrun conditions.

C. LATENCY

In hard and soft/firm real-time systems, *latency* is of primary importance, and it is required to strictly obey application-dependent deadlines. For this kind of systems, delays due to communication over a digital network must be bounded, and specific real-time industrial protocols and implementations are typically employed that offer some guarantees (under a statistical basis, for soft/firm real-time systems) for the worst-case transmission times [49], [50]. In TSCH, this can be achieved by exploiting deterministic channel access (i.e., time slotting) coupled with a proper scheduling of exchanges (matrix configuration). The theoretical *worst-case latency* (Max_d), assuming that the only source of delays are retransmissions, can be computed analytically as

$$\begin{aligned} \text{Max}_d &= N_{hop} \cdot N_{tries} \cdot T_{sframe} \\ &= N_{hop} \cdot N_{tries} \cdot (N_{slot} \cdot T_{slot}). \end{aligned} \quad (14)$$

Equation (14) only holds if, for every link in the path, no queuing of packets occurs inside any motes. A sufficient condition for this to happen, in the case all the cells reserved for the request/response exchange are not used by other traffic, is that $T_{app} \geq \text{Max}_d/N_{hop} = N_{tries} \cdot T_{sframe}$.

Conversely, when queuing phenomena take place, Max_d can be obtained by multiplying the value in (14) by the maximum size of the local queues (including the packet under transmission). From (14), it is clear that the worst-case latency can be reduced by lowering N_{slot} (which increases power consumption) or by lowering N_{tries} (which decreases power consumption but worsens reliability).

D. DERIVED QUANTITIES

We now change perspective by laying the ground for a comprehensive analysis of TSCH-based WSNs: starting from experimental results (packet losses and measured transmission latencies) we will show that a complete characterization of the network performance can be obtained in terms of reliability, power consumption, and latency. The basic idea is to firstly estimate ϵ from a set of experimental data by means of (4). Then, a number of intermediate quantities are derived, which permit the expected network behavior to be modelled and evaluated analytically.

$$\hat{N}_{tra}^{lost} = \hat{N}_{lost} \cdot \sum_{h=0}^{N_{hop}-1} \left[\frac{(1-\epsilon^{N_{tries}})^h - (1-\epsilon^{N_{tries}})^{h+1}}{1-(1-\epsilon^{N_{tries}})^{N_{hop}}} \right] \cdot \left(h \cdot \left(\frac{1}{1-\epsilon} - \frac{N_{tries} \cdot \epsilon^{N_{tries}}}{1-\epsilon^{N_{tries}}} \right) + N_{tries} \right) \quad (13)$$

Besides ϵ, which describes to what extent disturbance and interference impact on communication, the quantities that affect TSCH operation depend on:

- the characteristics of the protocol decided in the network design/configuration phase (N_{slot}, T_{slot}, and N_{tries});
- the characteristics of the path, which mainly depend on the physical placement of nodes and obstacles (N_{hop});
- the characteristics of the hardware of motes in terms of power consumption (E^f_{tx}, E^f_{rx}, and E^f_{listen});
- the characteristics of the software application that communicates over the network (T_{app} and N_{sam}).

One of the problems with the formerly presented analysis is that the measured number N_{lost} of failed requests in typical operating conditions may not describe disturbance and interference reliably. In fact, unless experiments are protracted for very long times (months), no packets are usually lost with the default retry limit. A more reliable estimate of the number of failed requests (for which no response is obtained) can be derived from (1) and corresponds to

$$\hat{N}_{lost} = N_{sam} \cdot \epsilon_{pkt} = N_{sam} \cdot \left(1 - (1 - \epsilon^{N_{tries}})^{N_{hop}}\right). \quad (15)$$

The expected number of transmission attempts performed for a packet correctly delivered on a single hop is described by a truncated geometric series

$$\frac{1}{1-\epsilon^{N_{tries}}} \sum_{k=1}^{N_{tries}} k(1-\epsilon)\epsilon^{k-1}.$$

Concerning the whole path, it can be easily demonstrated that an estimate $\hat{n}_{tra}$ of the average number of frames transmitted on air for a successful request/response exchange can be computed as

$$\hat{n}_{tra} = N_{hop} \cdot \left(\frac{1}{1-\epsilon} - \frac{N_{tries} \cdot \epsilon^{N_{tries}}}{1-\epsilon^{N_{tries}}} \right). \quad (16)$$

An estimate $\hat{N}^{lost}_{tra}$ of the overall number of frame transmission attempts associated to failed requests can be obtained from (13), where the factor in square brackets represents the fraction of packets that are lost in hop $h+1$ (that is, for which the preceding h hops were successful), while the factor in round brackets represents the average number of transmitted frames for each one of these packets (the first term resembles (16), while for the latter it should be remembered that, when a packet is lost because its transmission failed on a certain link, the number of attempts on that link equals N_{tries}).

An estimate of the rate f_{tra}, as defined by (8), can be obtained from the derived quantities $\hat{N}_{lost}$, $\hat{n}_{tra}$, and $\hat{N}^{lost}_{tra}$, and corresponds to

$$\hat{f}_{tra} = \frac{\hat{n}_{tra} \cdot (N_{sam} - \hat{N}_{lost}) + \hat{N}^{lost}_{tra}}{T_{app} \cdot N_{sam}}. \quad (17)$$

By substituting $\hat{f}_{tra}$ in (10), the estimate $\hat{f}_{listen}$ can be easily computed. In turn, the power consumption P is obtained from $\hat{f}_{tra}$ and $\hat{f}_{listen}$ by means of (7). The quantities E^f_{tx}, E^f_{rx} and E^f_{listen} for the commercial motes we used in our setup were evaluated experimentally, as described in the following section.

Finally, since the overall number of frames transmitted on air for a single end-to-end packet exchange equals N_{hop} when no errors occur, and every retry takes an additional slotframe, an estimate of the average transmission latency (that for request-response pairs coincides with the round-trip time) can be obtained from $\hat{n}_{tra}$ and d_{min} as

$$\hat{\mu}_d = d_{min} + \left(\frac{1}{2} + \hat{n}_{tra} - N_{hop}\right) \cdot T_{sframe}. \quad (18)$$

From a practical point of view, equations (15)–(17) are valid also in the presence of queuing phenomena. The only limitation is that they do not consider those packets that went lost due to the fact that the queue in the receiver (either an intermediate node or one of the end-points) was completely full at the time of arrival. This situation takes place seldom in real cases, and its occurrence is an indirect evidence that the network was not sized correctly. On the contrary, neglecting queuing delays as done in (18) constitutes an acceptable approximation only when (most of the times) packets manage to be forwarded at the earliest opportunity (i.e., in next suitable link). However, if the average delay experienced by packets in the queues of the traversed motes along the path can be estimated in some way, equation (18) can be updated to take into account this latency contribution as well.

After computing the value of ϵ by means of (4),[1] all the performance indicators we are interested in for describing a WSN based on TSCH from the point of view of applications, namely reliability $(1 - \epsilon_{pkt})$, power consumption (P), theoretical worst-case latency (Max_d), and average latency ($\hat{\mu}_d$), can be obtained analytically using equations (1), (7), (14), and (18), respectively.

IV. POWER CONSUMPTION MODEL

In this section, the power consumption for the specific motes we used in our experimental campaign was measured experimentally. Two main goals were reached: firstly, the real power consumption of OpenMote B devices was assessed; secondly, the actual values of E^f_{tx}, E^f_{rx} and E^f_{listen} were obtained. By substituting them in (7), an estimate of the actual power consumption for this kind of motes can be obtained, given the network traffic and TSCH configuration parameters.

A. OpenMote B DEVICES AND EXPERIMENTAL SETUP

All experiments were performed on OpenMote B devices [51]. These motes are equipped with a TI CC2538 System-On-Chip microcontroller, which integrates an ARM® Cortex™ M3 CPU with 32 KB of dynamic RAM memory and 512 KB of flash memory, and a radio transceiver compliant to IEEE 802.15.4 for transmission in the 2.4 GHz band. A second radio chip (Atmel AT86RF215), not used in our experiments, is also included on the electronic board of the motes, which

[1] The actual value of d_{min} can be also derived from the configuration of the TSCH matrix—see the example of Fig. 2.

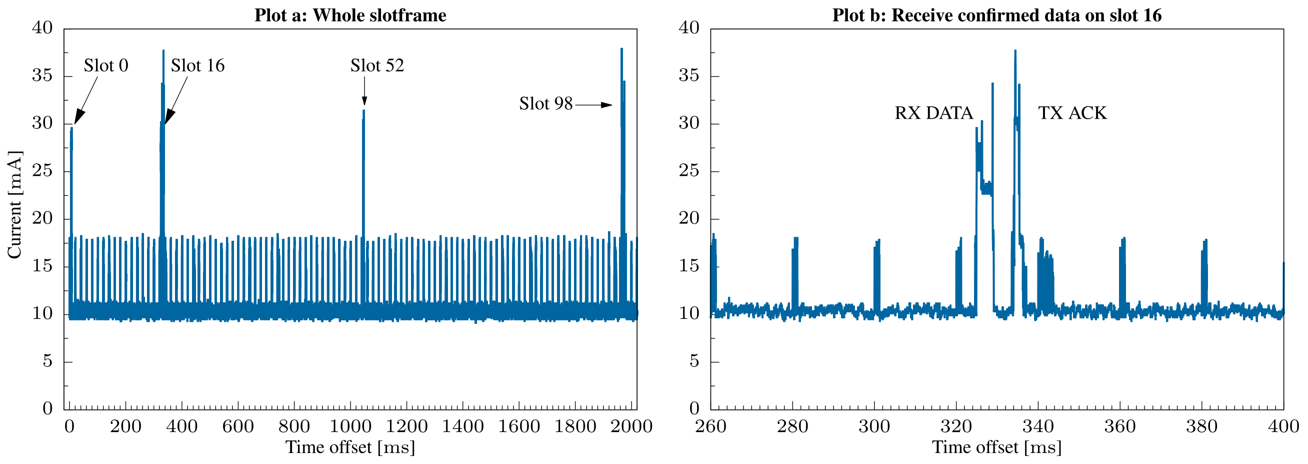


FIGURE 3. Power consumption for the whole duration of the slotframe (Plot 3.a) and for a zoomed out portion of it that embraces seven slots (Plot 3.b). In the plot of the right side, the reception of a confirmed frame (bearing a `ping` request) can be observed in the fourth slot.

manages sub-GHz transmissions (868/915 MHz). In addition, these motes are equipped with temperature and relative humidity sensors, and four indicator LEDs. Typically, such a kind of devices are operated on batteries, but they can also be powered by means of a conventional USB cable. From a software point of view, the OpenWSN [52] and the Contiki operating systems are supported. Both of them are available as open source, and so their code can be easily inspected and modified, if needed.

All the experimental campaigns we describe in this article are based on OpenWSN (version REL-1.24.0), as it includes the most recent version of the 6TiSCH protocol stack. In particular, measurements for both power consumption and latency (this latter reported in Section V) were carried out using a simple network setup, made up of two OpenMote B devices (the root plus a leaf mote acting as a responder) located about 2 m apart. The root mote was connected to a PC running the Linux operating system, on which OpenVisualizer (the network management software distributed along with OpenWSN) was executed. This tool was used to select the root mote, as well as to control and monitor the state of the WSN. Samples about round-trip times and packet losses were collected on the PC by periodically invoking the `ping` command to query the leaf mote.

B. CHARACTERIZATION OF POWER CONSUMPTION

The power consumption of OpenMote B motes was estimated by means of a Tektronix MDO3024 oscilloscope equipped with two TPP0250 probes. The probes were connected to two specific pins of the printed circuit board of the mote that are devoted to measuring the total current flow it absorbs. Current absorption was measured only on the leaf mote, which is the one that, in real applications, is powered on batteries. The OpenWSN code was modified to disable the onboard LEDs (very power-hungry), so that all the experimental results were obtained with such indicators switched off. This led to a reduction of the absorbed current equal, on average, to about 6 mA, corresponding to about 360 μJ saved for every slot.

The experimental measures of the current drained by the leaf mote over one slotframe are reported in Fig. 3.a, where the x-axis represents the offset (in milliseconds) from the beginning of the slotframe. The many regularly-spaced short peaks are related to activities performed by the protocol stack at the beginning of every slot. Consequently, their number is equal to $N_{slot} = 101$. Fig. 3.b zooms out part of the diagram in Fig. 3.a, and includes seven peaks, which clearly appear to occur every $T_{slot} = 20$ μs. The two current consumption patterns labeled *RX DATA* and *TX ACK*, which rise above the previous peaks, refer to a confirmed transmission (associated to a `ping` request/response exchange) as seen by the point of view of the leaf mote that manages the RX cell.

For all the experiments described in this section, the cells in the TSCH matrix are configured as follows:

- Slot 0 (time offset 0 ms) is a shared TXRX cell, which is typically associated with channel offset 0. It is used to send frames related to network formation and maintenance (e.g., enhanced beacons).
- Slot 16 (time offset 320 ms) is configured in the root as a shared TXRX cell, while from the point of view of the responder it behaves as a dedicated RX cell. After the initial network configuration phase, it is typically used by the root only to send confirmed packets to its children in the downward direction, hence collisions may not happen. In the context of this article, this cell is used for transferring `ping` request packets. An example that shows what happens to current absorption in the responder during the reception of a frame bearing the `ping` request and the transmission of the related ACK frame can be found in Fig. 4.b. When no frame is transferred and the cell remains unused, the responder still spends a considerable amount of energy to listen to the channel (idle listening), as shown in Fig. 4.c.
- Slot 98 (time offset 1960 ms) is configured in the responder as a non-shared TX cell, and is reserved for transmission to its parent (the root) in the upward direction. In the experiments, this cell is used for transferring packets

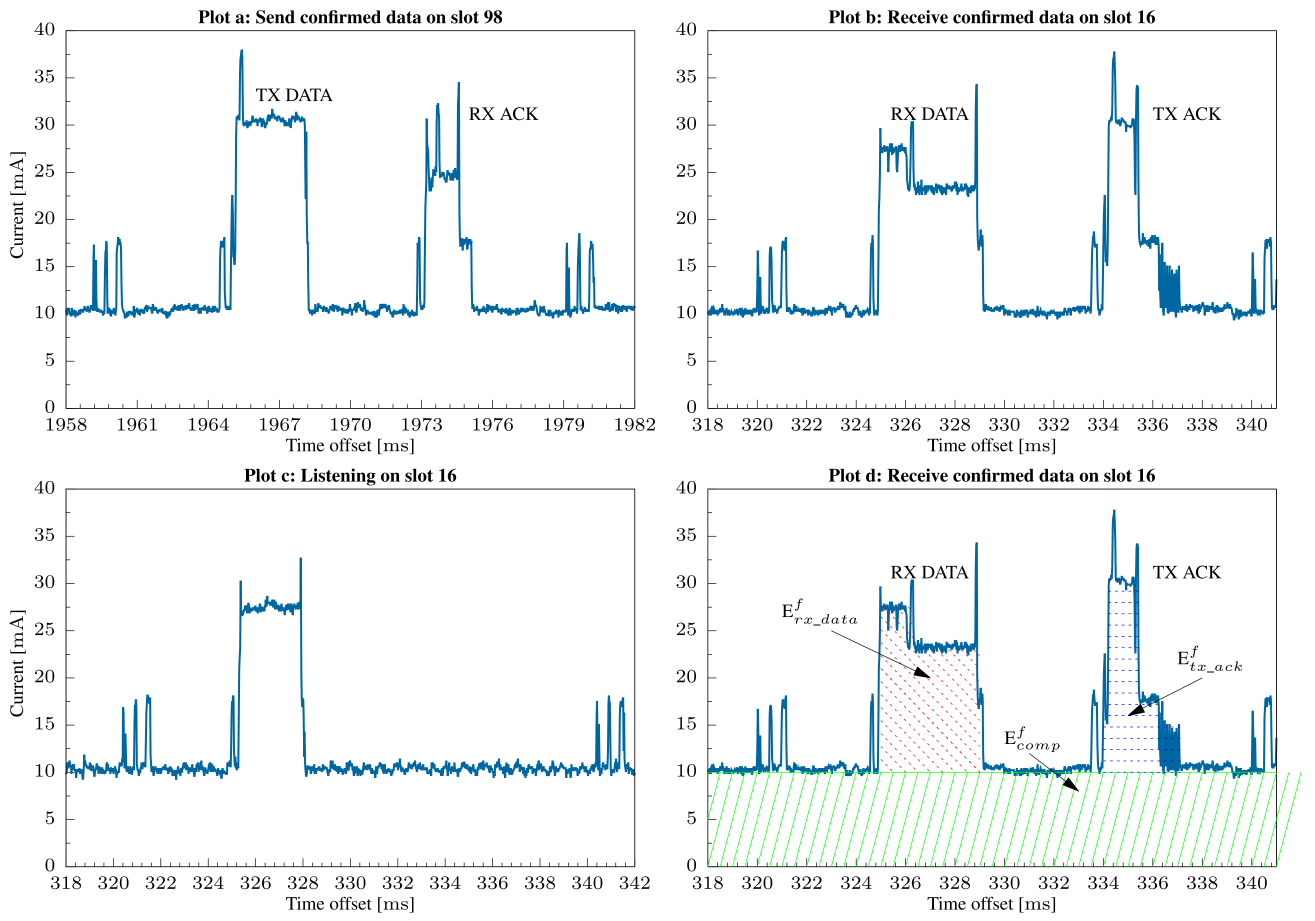


FIGURE 4. Power consumption for the different types of cells: slot including a confirmed frame transmission (a), slot including confirmed frame reception (b), slot in which idle listening occurs (c), and dissection of a confirmed frame reception into separate contributions (d).

TABLE 3. Energy consumption for different types of actions within a slotframe matrix with OpenMote B motes. In bold quantities used in Eq. (7).

Quantity	Action(s)	Slot offset	Energy [μJ]	Size [bytes]
$E^f_{rx\_data}$	RX DATA frame	16	178	87
$E^f_{tx\_ack}$	TX ACK frame	16	106	33
$E^f_{tx\_data}$	TX DATA frame	98	187	90
$E^f_{rx\_ack}$	RX ACK frame	98	79	33
E^f_{listen}	Idle listening	16	**138**	-
E^f_{comp}	Computation	-	628	-
E^f_{rx}	RX DATA + TX ACK	16	**284**	120
E^f_{tx}	TX DATA + RX ACK	98	**266**	123

bearing the `ping` response, an example of which is shown in Fig. 4.a.

- Slot 52 (time offset 1040 ms) is configured in the responder as a shared TXRX cell, and is devoted to communication with its children. However, as this mote in our setup is a leaf (i.e., it has no children) the cell remained unused in the experiment.

Experimental results related to power consumption are reported in Table 3. They were obtained by performing a numerical integration on the experimental samples included in the above plots (plus other samples not reported for space reasons), and by multiplying the resulting area, which refers to an overall electric charge in microcoulombs (μC), by the supply voltage of the mote (3 V), in order to obtain the power consumption in microjoule (μJ).

In particular, the quantity E^f_{comp} refers to the power consumption when no transmission or reception are performed within the slot, i.e., the global consumption of the mote excluding operations of the network component. Such a quantity corresponds to the rectangular area at the bottom of Fig. 4.d filled with a continuous-green-lines pattern and, for a time slot lasting 20 ms, it is about 628 μJ. This value is noticeably higher than for other kinds of motes. For instance, in [53] a similar analysis was performed for motes based on the STM32F103RB 32-bit microcontroller and the Atmel AT86RF231 radio chip. In that case, the power consumption (obtained by multiplying the charge in microcoulomb reported in the paper by 3 V) was about 113.4 μJ. The abnormally high power consumption we obtained in our setup depends essentially on the fact that OpenMote B devices, under several respects, are still prototypes, and the software they run (OpenWSN) is not optimized for energy saving yet (e.g., by switching the CPU to deeper sleep states when no operations are needed).

TABLE 4. Experimental results about the influence of N_{slot} on latency, reliability, and power consumption (measures on real devices).

	Latency								Reliability			Power Consumption				
N_{slot}	d_{min}	μ_d	σ_d [s]	d_{p99}	d_{max}	$\hat{n}_{tra}$ [#]	$\hat{\mu}_d$ [s]	Max$_d$ [s]	P_{lost}	ϵ	$1-\epsilon_{pkt}$	f_{tra} [$\cdot 10^{-5}$]	f_{listen} [$\cdot 10^{-4}$]	P μW	$\hat{f}_{tra}$ [$\cdot 10^{-5}$]	$\hat{f}_{listen}$ [$\cdot 10^{-4}$]
11	0.212	0.409	0.194	1.231	1.438	2.34	0.399	7.040	0.0	0.148	12-nines	2.00	90.70	1262.8	1.95	90.71
31	0.491	0.982	0.431	2.301	3.419	2.27	0.969	19.840	0.0	0.119	14-nines	1.91	32.06	453.0	1.89	32.06
51	0.258	1.024	0.649	3.007	3.054	2.25	1.021	32.640	0.0	0.110	15-nines	1.88	19.41	278.3	1.87	19.42
91	0.497	1.741	1.046	4.861	5.397	2.25	1.858	58.240	0.0	0.110	15-nines	1.87	10.80	159.4	1.87	10.80
101	0.352	2.046	1.588	8.764	10.457	2.28	1.936	64.640	0.0	0.124	14-nines	1.95	9.70	144.7	1.90	9.71
151	2.877	5.036	1.755	8.846	14.557	2.25	5.135	96.640	0.0	0.110	15-nines	1.85	6.44	99.0	1.87	6.43
201	0.726	4.216	2.880	12.131	14.050	2.36	4.193	128.640	0.0	0.153	12-nines	1.97	4.78	76.7	1.97	4.78

The other quantities reported in the table refer to the network component only. For instance, the values of $E^f_{rx\_data}$ and $E^f_{tx\_ack}$ were obtained through numerical integration of the areas filled with red-dashed-lines and blue-horizontal-dashed-lines patterns in Fig. 4.d, respectively. Starting from the consumption related to DATA and ACK frames, the quantities $E_{rx} = E^f_{rx\_data} + E^f_{tx\_ack}$ and $E_{tx} = E^f_{tx\_data} + E^f_{rx\_ack}$ can be obtained, which, for a confirmed frame exchange, correspond to the consumption on the receiver side (RX DATA + TX ACK) and on the transmitter side (TX DATA + RX ACK), respectively.

To emulate application-level request/response exchanges (as those performed by CoAP), we used the `ping` utility. In all the experiments, the size of the payload included in `ping` packets was set to 30 bytes. These packets are encoded using the rules of the IEEE 802.15.4 standard, which leads to the frame sizes reported in Table 3 (that also include the physical preamble). Since in our setup we considered the responder (leaf node), E_{rx} is associated to an ICMP *echo request* packet while E_{tx} is associated to an ICMP *echo reply* packet.

V. EXPERIMENTAL RESULTS

Two sets of experiments were performed to analyze the effects of the most relevant protocol parameters on reliability, power consumption, and latency. In particular, we evaluated what happens when either the length of the slotframe or the maximum number of allowed tries are varied.

The WSN under test is composed of two nodes (OpenMote B devices), and is exactly the same as the setup used for evaluating power consumption in Section IV-A. This overly simplified configuration does not limit validity of results. In fact, concerning data exchanges, the TSCH matrix permits to partition the whole traffic into independently managed cells according to a known and configurable schedule. The only additional complexity when dealing with multi-level networks is that, due to the wider coverage of the network, the frame error probability ϵ on distinct links may differ, and (4) only provides an aggregated value for it. The `ping` generation period (that corresponds to T_{app}) was set to 120 s, and every experiment lasted 4 hours. Therefore, the number of samples collected in each experiment is $N_{sam} = 120$.

To emulate a harsh environment, four interfering IEEE 802.11 stations were placed near the two OpenMote B devices to inject a configurable amount of traffic on air, which adds to the background traffic generated by Wi-Fi networks deployed in nearby offices and labs. So that the injected traffic can affect all the 16 channels defined by IEEE 802.15.4 in the ISM band (from 2.405 to 2.480 GHz), remembering that the width of an OFDM channel in IEEE 802.11 is 20 MHz, we configured four access points (AP) to operate on channels 1 (2.412 GHz), 5 (2.432 GHz), 9 (2.452 GHz) and 13 (2.472 GHz), respectively, and associated each interfering station to a different AP. Details about inner operation of interferers can be found in [17]. Basically, every interfering station executes a finite state machine with two states, *inactive* and *active*. In the *active* state a burst of packets is generated. The number of packets within the burst is selected randomly according to a truncated exponential distribution: on average, 100 packets (but no more than 500) are sent. Packets within the burst are generated periodically with period 400 μs, and their size is 1500 B. After the burst the interfering station enters the *inactive* state, where it stops transmitting for a random time (gap) whose duration follows a truncated exponential distribution with mean 560 ms. The maximum duration of the *inactive* state is limited to 20 s.

A. PERFORMANCE VS. SLOTFRAME LENGTH

In the first set of experiments we analyzed the impact of the number N_{slot} of slots in a slotframe on the performance indicators analyzed in Section III. In particular, this parameter was varied between 11 and 201 (the default value in OpenWSN is $N_{slot} = 101$). Motes were restarted at the beginning of every experiment. This is unavoidable, because modifying N_{slot} requires the operating system to be recompiled and reloaded on every mote, which implies a new network formation and hence a new TSCH matrix. For this reason, each experiment is characterized by a different value of d_{min}. The other parameters of the network were kept at their default value, and in particular $N_{tries} = 16$.

Results of the experimental campaign are reported in Table 4. As expected, smaller values of N_{slot} improve network responsiveness: if $N_{slot} = 11$ the measured maximum latency is 1.438 s, whereas for $N_{slot} = 201$ it grows up

TABLE 5. Experimental results about the influence of N_{tries} on latency, reliability, and power consumption (measures on real devices).

	Latency								Reliability			Power Consumption				
N_{tries}	d_{min}	μ_d	σ_d [s]	d_{p99}	d_{max}	$\hat{n}_{tra}$ [#]	$\hat{\mu}_d$ [s]	Max_d [s]	P_{lost}	ϵ	$1-\epsilon_{pkt}$	f_{tra} $[\cdot 10^{-5}]$	f_{listen} $[\cdot 10^{-4}]$	P μW	$\hat{f}_{tra}$ $[\cdot 10^{-5}]$	$\hat{f}_{listen}$ $[\cdot 10^{-4}]$
2	0.496	1.851	1.015	4.441	5.377	2.17	1.861	8.080	0.017	0.0963	0.98154	1.82	9.71	144.1	1.82	9.71
4	0.342	1.853	1.272	6.066	6.090	2.24	1.850	16.160	0.0	0.1102	0.99971	1.88	9.71	144.3	1.87	9.71
6	0.387	2.031	1.323	6.906	7.447	2.32	2.048	24.240	0.0	0.1388	0.99999	1.93	9.70	144.5	1.93	9.70
8	0.726	2.320	1.558	8.255	9.890	2.27	2.285	32.320	0.0	0.1197	7-nines	1.92	9.70	144.5	1.89	9.71
16	0.352	2.046	1.588	8.764	10.457	2.28	1.936	64.640	0.0	0.1244	14-nines	1.95	9.70	144.6	1.90	9.71

to 14.050 s (theoretical worst-case values are 7.04 s and 128.64 s, respectively). On the other hand, power consumption when $N_{slot} = 11$ is about 150 times higher than when $N_{slot} = 201$ (1262.8 μW and 76.7 μW, respectively). This means that, concerning the selection of N_{slot}, a compromise has to be found.

Above behavior is due to the fact that, although the actual rate f_{tra} of frame transmissions on air did not vary appreciably for the different experiments, there is an increase of the rate f_{listen} with which motes listen to the network for receiving frames. Basically, f_{tra} remains stable because it depends on the (fixed) packet generation period at the application level and the quality of communication on the wireless medium (i.e., the frame error probability ϵ). As expected, the values of ϵ estimated in the different experiments are not exactly the same. In fact, the traffic injected by the interfering Wi-Fi nodes was, on average, fixed, but the load caused by other Wi-Fi devices located close to the motes was out of our control and may vary unpredictably. The values obtained for ϵ ranged from 11.0 % and 15.3 %. In spite of this variation, because of the high value set for the retry limit the measured *packet loss ratio* $P_{lost} = N_{lost}/N_{sam}$ was equal to 0 in all the experiments.

Values for $\hat{\mu}_d$, $\hat{f}_{tra}$, and $\hat{f}_{listen}$ in Table 4 are obtained starting from ϵ and derived quantities (that is, not measured directly in the experiments), and can be used to cross-check the model. Although experiments included only 120 samples, they are very close to the corresponding measured quantities, i.e., μ_d, f_{tra}, and f_{listen}.

B. PERFORMANCE VS. RETRY LIMIT

In the second set of experiments we varied N_{tries} between 2 and 16. This parameter is mostly related to reliability, but enlarging its value worsens latency and power consumption (even though they are affected to a lower extent). For N_{slot} we used 101, i.e., the default value. Results are reported in Table 5. The probability $1 - e^{pkt}$ that a packet is successfully delivered to destination (reliability) quickly approaches 1. When $N_{tries} = 2$, $1 - e^{pkt} = 0.98154$, while reliability is as high as 0.99999999999993 (14-nines) when N_{tries} is increased to 16. As expected, we managed to measure a number of lost packets different from 0 only by setting $N_{tries} = 2$. In this case, the measured packet loss ratio is $P_{lost} = 0.017$, which is quite close to the $1 - e^{pkt}$ estimate.

Increasing the retry limit had limited effects on latency and power consumption, because the frame error probability ϵ in our setup was not particularly high (about $10 - 15\%$). This implies that the probability to perform many repetitions in a row is small, as demonstrated by the fact that $1 - e^{pkt}$ quickly converges to 1. As can be seen from the table, all statistics that refer to the latency (including μ_d, σ_d and d_{p99}) are negatively affected by an increase of the retry limit. It can be observed the good match between the mean value μ_d of the measured latency and its expected value $\hat{\mu}_d$. Also the measured maximum latency d_{max} increased from 5.377 s to 10.457s. However, it is worth noting that this quantity is not statistically reliable because of the limited number of samples. In fact, the worst-case latency Max_d ranges in theory from 8.080s when $N_{tries} = 2$ to 64.640 s when $N_{tries} = 16$.

The effects of N_{tries} on power consumption are mostly negligible. In particular, P lay in the range between 144.1 μW and 144.6 μW. This is due to two reasons: first, with the parameters we selected for the network and the application, consumption mainly depends on idle listening; second, multiple retries are only performed when frame transmission repeatedly fails, which is an unlikely event in our setup.

VI. PRACTICAL APPLICATION CONTEXTS

The results provided by the network model were found to be comparable with measurements obtained from real devices. Hence, the model can be profitably used alone (i.e., without a real setup) to better investigate the effect of parameters N_{slot} and N_{tries} on performance indicators in specific operating conditions. As the final part of this article, four experiments were carried out to assess the performance of a real setup in specific application contexts that demand high reliability, low latency, and low power consumption, respectively

A. LEVERAGING THE MATHEMATICAL MODEL

The model was used to analyze the behavior of TSCH in the case the frame error probability ϵ is increased up to 0.4, this denoting a quite hostile environment. The value of d_{min} depends on the configuration of the TSCH matrix, but for our analysis it is not so relevant, because it only causes a shift on the time-axis (latency offset). For this reason, we set $d_{min} = 500$ ms for all the plots shown in the figure.

Initially, the number of slots in a slotframe was varied between 11 and 201. Results are reported in Plot 1 of

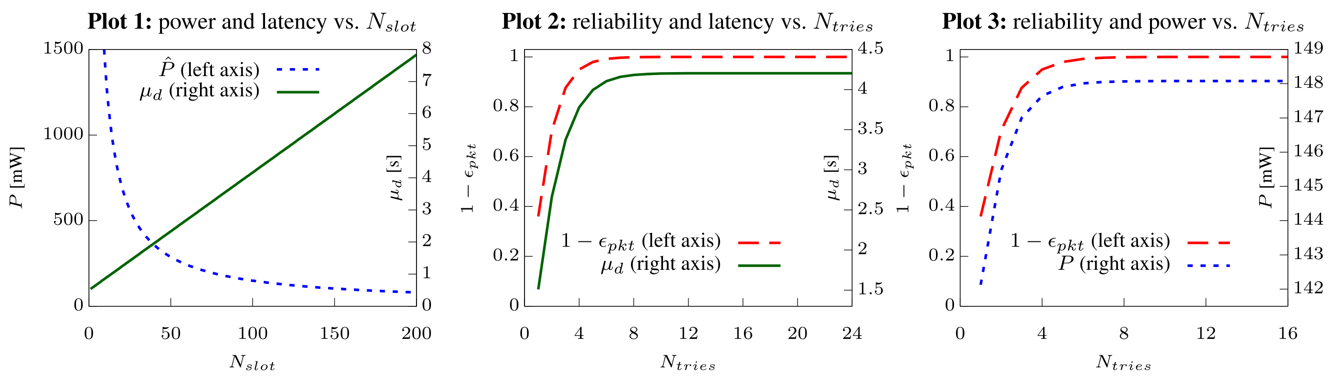


FIGURE 5. Influence of N_{slot} and N_{tries} on reliability, power consumption, and latency, evaluated using the proposed network model ($\epsilon = 0.4$, $N_{tries} = 16$ for Plot 1, $N_{slot} = 101$ for Plot 2 and Plot 3).

Fig. 5. For calculating P we used (7), where the value $\hat{f}_{tra}$ is obtained from (17). As expected, the average latency, obtained from (18), increases linearly with N_{slot}, while power consumption decreases according to an inverse proportionality law. When N_{slot} is greater than ~ 58, power consumption falls below 250 μW. At the same time, the average latency exceeds 2.6 s. Decreasing N_{slot} well below such threshold allows a consistent reduction of latency but, due to the increase in power consumption, motes should be probably connected to an external power supply.

Plots 2 and 3 of Fig. 5 are obtained by varying N_{tries} from 1 to 16. As highlighted in both graphs, reliability $1-\epsilon_{pkt}$, given by (1), increases as N_{tries} grows. A similar trend, but with some relevant differences, can be observed for the average latency μ_d in Plot 2 and the power consumption P in Plot 3. In both cases the shape of the plot is seemingly the same but, in terms of the magnitude of the increase, the ranges in which values vary are noticeably different. The average latency is influenced more, and increases from 1.5 s when $N_{tries} = 2$ to values slightly above 4 s when $N_{tries} \geq 5$. Instead, in the very same conditions, power consumption grows from 142 mW to 148 mW, which is indeed a modest increase. This is due to the fact that, in the considered operating conditions, the most part of energy is spent for idle listening and initial transmission attempts (including the very first retries possibly performed for each frame). On the contrary, only a limited contribution is related to the additional retransmissions made available by increasing the retry limit, since a small fraction of frames experience many repeated failures.

Summing up, increasing N_{slot} has a positive effect on power consumption, a negative effect on latency, and no effects on reliability. Instead, increasing N_{tries} has a positive effect on reliability, a negative effect on latency, and a slight negative effect on power consumption (practically negligible for the typical values of the frame error probability). In the following, some working points of interest were identified from the previous plots and analyzed by means of an extensive experimental campaign on real hardware.

B. EVALUATION OF RELEVANT CONFIGURATIONS

As stated in the introduction, WSNs are exploited in a plurality of application contexts with very different (and often conflicting) requirements. We identified four sample configurations for a TSCH network, each of which suits the needs of some specific context. They are denoted *default*, *high reliability*, *low latency*, and *low power consumption*, and are characterized by specific values for N_{slot} and N_{tries}.

The *default* configuration refers to out-of-the-box OpenWSN devices. To find suitable values for the network parameters of the other configurations, we started from the plots in Fig. 5, re-evaluated for $\epsilon = 0.13$, which more closely resembles the frame error probability typically observed in our experimental environment. Results are reported in Fig. 6. Then, three reasonable working points were sought on the plots, aimed to optimize every one of the performance indicators (reliability, latency, or power consumption) without penalizing excessively the others.

The effect of separately increasing/decreasing each parameter (either N_{slot} or N_{tries} is varied in every plot) is shown using underlined labels. It is worth remarking that working points in the figure and configurations are not the same. For example, the working point in Plot 4 labeled "lower latency", obtained by reducing N_{slot} from 101 to 11, does not correspond to the *low latency* configuration. In fact, in the latter case the value of N_{tries} was additionally decreased from 16 to 3, to lower the maximum latency further—see equation (14). As will be shown, there is not an optimal setting for N_{slot} and N_{tries} that suits all application contexts, because improving a performance indicator unavoidably worsens at least one of the others.

In detail, the four configurations we considered are:

- *Default* ($N_{slot} = 101$, $N_{tries} = 16$): this configuration represents the default out-of-the-box setting of a WSN when motes are based on the OpenWSN operating system, and constitutes a balanced trade-off among power consumption, reliability, and latency. In the following, it will be used as the baseline.
- *High reliability* ($N_{slot} = 101$, $N_{tries} = 24$): this configuration is characterized by a reliability level higher than the default case. In particular, 8 additional retries were granted to every frame transmission (N_{tries} is increased from 16 to 24).

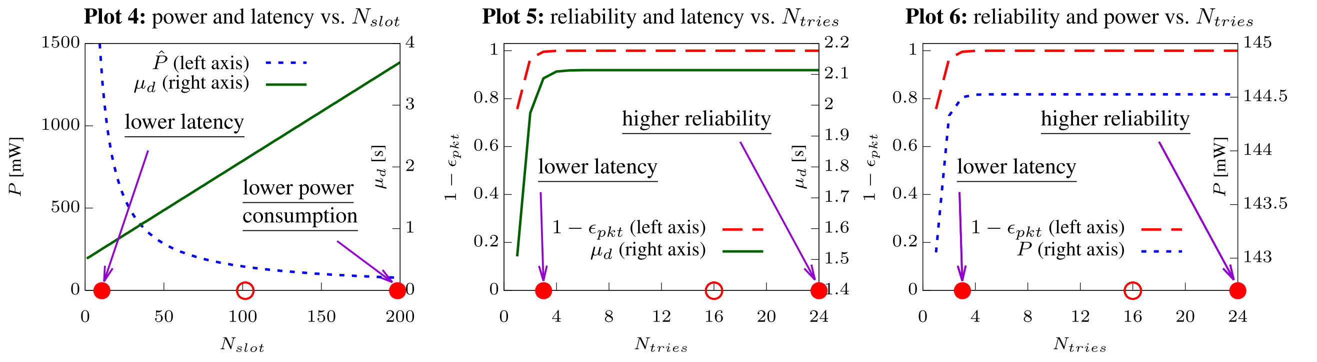


FIGURE 6. Influence of N_{slot} and N_{tries} on reliability, power consumption, and latency, evaluated using the proposed network model ($\epsilon = 0.13$, $N_{tries} = 16$ for Plot 4, $N_{slot} = 101$ for Plot 5 and Plot 6). Effects of moving working points—marked with solid red circles (●)—away from the *default* configuration—marked with empty red circles (○)—are suitably labeled.

TABLE 6. Latency, reliability, and power consumption, measured on real devices, related to four configurations (characterized by different values of N_{slot} and N_{tries}) targeted to different application contexts.

Configuration			Latency						Reliability			Power Consumption		
Condition	N_{slot}	N_{tries}	d_{min}	μ_d	σ_d [s]	d_{p99}	d_{max}	Max_d [s]	P_{lost}	ϵ	$1-\epsilon_{pkt}$	f_{tra} $[\cdot 10^{-5}]$	f_{listen} $[\cdot 10^{-4}]$	P μW
Default	101	16	0.528	2.115	1.310	6.579	11.049	64.640	0.0	0.125	14-nines	1.91	9.71	144.4
High Reliability	101	24	1.470	3.090	1.320	7.450	9.360	96.960	0.0	0.132	20-nines	1.92	9.71	144.5
Low Latency	11	3	0.159	0.336	0.135	0.780	1.023	1.320	0.0042	0.142	0.9942	1.92	90.71	1262.4
Low Power Cons.	201	16	2.565	5.535	2.461	13.637	22.366	128.640	0.0	0.112	14-nines	1.92	4.78	76.5
Default (15-days)	101	16	0.522	2.114	1.289	6.393	12.382	64.640	0.0	0.126	14-nines	1.90	9.71	144.5

- *Low latency* ($N_{slot} = 11$, $N_{tries} = 3$): this configuration is targeted to applications demanding low latency. In this case, N_{slot} was reduced by one order of magnitude (from 101 to 11), which lowers all statistical indices about latency (including the average and maximum values), and N_{tries} was set to 3 to provide stricter bounds on the worst-case latency.
- *Low power consumption* ($N_{slot} = 201$, $N_{tries} = 16$): this configuration is conceived for those applications where batteries on motes have to be replaced as seldom as possible. To this purpose, the value of N_{slot} was doubled from 101 to 201.

An experimental campaign was carried out on real OpenMote B devices for the four above configurations, using the network setup described in Section IV-A. The duration of each experiment was set to 24 hours ($N_{sam} = 720$), and the purposely injected interfering traffic was the same as in the previous experiments. Results are reported in Table 6.

The *high reliability* configuration (second row in the table) is clearly characterized by an extremely low packet loss probability. Under the hypothesis that frame attempts can be modeled as Bernoulli trials, reliability $1 - e^{pkt}$ is, in theory, as high as 0.99999999999999999999983, which corresponds to 20-nines. Although this parameter setting implies a tangible increase of the theoretical worst-case latency Max_d with respect to the default case, both the measured latency and the power consumption were only marginally affected, as the transmission error ϵ on the channel is not excessively high.

Concerning the *low latency* configuration, all statistics about latency are noticeably lower than the default case. In particular, the measured maximum ($d_{max} = 1.023$ s) is very small and quite close to the theoretical worst-case value ($\text{Max}_d = 1.320$ s), while the average time taken to perform a request/response exchange is $\mu_d = 0.336$ s. Unfortunately, low latency settings led to a significant increase of the measured packet losses ($P_{lost} = 0.0042 = 0.42$ %) and to a decrease of the estimated reliability ($1 - e^{pkt} = 0.9942 = 99.42$ %). As expected, power consumption is about 9 times higher than the default case, which makes this solution mostly unpractical for battery-powered devices.

Finally, with the *low power consumption* configuration, the power consumed by motes dropped to 76.5 μW, which is about half than in the default case (144.4 μW). This is clearly due to the fact that receiving motes have to switch their radio transceivers on (to listen to the channel) less frequently. In particular, since the actual rate of packet transmissions does not vary (it only depends on the sending period T_{app} of the application), f_{listen} decreased from 9.71 to 4.78 occurrences of idle listening per seconds. The main disadvantage of this setting is that there is a sharp increase of the latency.

A swift characterization of the above configurations based on power consumption, reliability, and latency, is outlined in the radar charts in Fig. 7, which clearly highlight the related application contexts. Green points represent the quantity that mainly benefits from a given setting, while red points are the effects (almost always negative) that such setting implies on the other two quantities. As can be seen, optimizing all performance indicators at the same time is not possible.

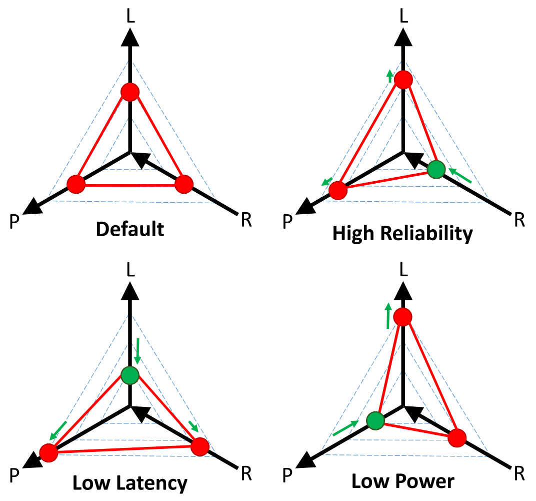


FIGURE 7. Effects of different parameter configurations (targeted to specific application contexts) on reliability (R), latency (L), and power consumption (P).

To check the reliability of the statistics we obtained from experimental samples, performance evaluation for the default configuration was left running after the first 24 hours, so that the experiment lasted, on the whole, 15 days. By doing so, $N_{sam} = 10800$ samples were actually collected (including those used to calculate the values in the first row of Table 6). Results are reported in the last row of the table. Also in this case, no packets went lost, not even after two weeks of continuous operation, as witnessed by the fact that all `ping` requests succeeded. Measured latency was bounded to a maximum of 12.382 s and the 99-percentile was 6.393 s.

Results for this extended data set show a good match with those obtained over a single day, which confirms that statistics of the latter are reliable enough.

VII. CONCLUSION

In this article, a mathematical model of a TSCH-based network has been proposed, which provides three performance indicators that are relevant for WSNs: reliability, power consumption, and latency. This model relies on a few parameters that can be easily estimated from a real setup deployed in the area of interest. These parameters can then be used in a later stage to evaluate the expected network performance when some TSCH configuration parameters are varied. To provide a realistic estimation of the power consumption, we carried out a characterization of the latest version of OpenMote B motes based on measurements performed on a real setup, and the related results have been presented here.

The analysis of three relevant application contexts, characterized by specific network configuration parameters (slotframe duration and retry limit) highlighted that reliability, power consumption, and latency are intimately connected. Therefore, it is not possible to optimize all the performance indicators at once, but their joint optimization must necessarily follow a holistic approach. TSCH proves to be a quite flexible solution, as the performance demanded by a wide range of application contexts can be easily achieved by suitably tuning these parameters. As an example we showed that, for a two-way communication where every packet performs two hops on the whole (both directions considered), in typical operating conditions it is possible to either decrease the average latency below $\sim \frac{1}{3}$ s, or ensure 20-nines reliability, or, finally, reduce the power consumption of the network component by half with respect to the default configuration.

REFERENCES

[1] J. Horneber and A. Hergenröder, "A survey on testbeds and experimentation environments for wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1820–1838, 4th Quart., 2014.

[2] B. Rashid and M. H. Rehmani, "Applications of wireless sensor networks for urban areas: A survey," *J. Netw. Comput. Appl.*, vol. 60, pp. 192–219, Jan. 2016.

[3] X. Yu, P. Wu, W. Han, and Z. Zhang, "A survey on wireless sensor network infrastructure for agriculture," *Comput. Standards Interfaces*, vol. 35, no. 1, pp. 59–64, Jan. 2013.

[4] S.-E. Yoo, P. K. Chong, D. Kim, Y. Doh, M.-L. Pham, E. Choi, and J. Huh, "Guaranteeing real-time services for industrial wireless sensor networks with IEEE 802.15.4," *IEEE Trans. Ind. Electron.*, vol. 57, no. 11, pp. 3868–3876, Nov. 2010.

[5] C.-S. Chen and D.-S. Lee, "Energy saving effects of wireless sensor networks: A case study of convenience stores in taiwan," *Sensors*, vol. 11, no. 2, pp. 2013–2034, Feb. 2011.

[6] M. Erdelj, M. Król, and E. Natalizio, "Wireless sensor networks and multi-UAV systems for natural disaster management," *Comput. Netw.*, vol. 124, pp. 72–86, Sep. 2017.

[7] B. Velusamy and S. C. Pushpan, "An enhanced channel access method to mitigate the effect of interference among body sensor networks for smart healthcare," *IEEE Sensors J.*, vol. 19, no. 16, pp. 7082–7088, Aug. 2019.

[8] G. Acar and A. E. Adams, "ACMENet: An underwater acoustic sensor network protocol for real-time environmental monitoring in coastal areas," *IEEE Proc.-Radar, Sonar Navigat.*, vol. 153, no. 4, pp. 365–380, Aug. 2006.

[9] I. C. Chimsom and M. K. Habib, "Design of a two-tier WSN-based IoT surveillance system with cloud integration," in *Proc. 20th Int. Conf. Res. Edu. Mechatronics (REM)*, May 2019, pp. 1–7.

[10] Z. Zhang, A. Mehmood, L. Shu, Z. Huo, Y. Zhang, and M. Mukherjee, "A survey on fault diagnosis in wireless sensor networks," *IEEE Access*, vol. 6, pp. 11349–11364, 2018.

[11] Y. Zhang and W. W. Li, "Energy consumption analysis of a duty cycle wireless sensor network model," *IEEE Access*, vol. 7, pp. 33405–33413, 2019.

[12] H. Yetgin, K. T. K. Cheung, M. El-Hajjar, and L. Hanzo, "A survey of network lifetime maximization techniques in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 828–854, 2nd Quart., 2017.

[13] M. Huang, K. Zhang, Z. Zeng, T. Wang, and Y. Liu, "An AUV-assisted data gathering scheme based on clustering and matrix completion for smart ocean," *IEEE Internet Things J.*, early access, Apr. 15, 2020, doi: 10.1109/JIOT.2020.2988035.

[14] Z. Li, Y. Liu, A. Liu, S. Wang, and H. Liu, "Minimizing convergecast time and energy consumption in green Internet of Things," *IEEE Trans. Emerg. Topics Comput.*, vol. 8, no. 3, pp. 797–813, Jul. 2020.

[15] M. Peng, W. Liu, T. Wang, and Z. Zeng, "Relay selection joint consecutive packet routing scheme to improve performance for wake-up radio-enabled WSNs," *Wireless Commun. Mobile Comput.*, vol. 2020, pp. 1–32, Jan. 2020.

[16] G. Cena, S. Scanzio, and A. Valenzano, "Improving effectiveness of seamless redundancy in real industrial Wi-Fi networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2095–2107, May 2018.

[17] G. Cena, S. Scanzio, and A. Valenzano, "Experimental evaluation of techniques to lower spectrum consumption in wi-red," *IEEE Trans. Wireless Commun.*, vol. 18, no. 2, pp. 824–837, Feb. 2019.

[18] V. N. Swamy, P. Rigge, G. Ranade, B. Nikolić, and A. Sahai, "Wireless channel dynamics and robustness for ultra-reliable low-latency communications," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 4, pp. 705–720, Apr. 2019.

[19] G. Cena, S. Scanzio, L. Seno, and A. Valenzano, "Comparison of mixed diversity schemes to enhance reliability of wireless networks," in *Ad-Hoc, Mobile, and Wireless Networks*, M. R. Palattella, S. Scanzio, and S. C. Ergen, Eds. Cham, Switzerland: Springer, 2019, pp. 118–135.

[20] D. De Guglielmo, B. Al Nahas, S. Duquennoy, T. Voigt, and G. Anastasi, "Analysis and experimental evaluation of IEEE 802.15.4e TSCH CSMA-CA algorithm," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1573–1588, Feb. 2017.

[21] *IEEE Standard for Low-Rate Wireless Networks*, IEEE Standard 802.15.4-2015 (Revision of IEEE Standard 802.15.4-2011, Apr. 2016, pp. 1–709.

[22] A. Nikoukar, S. Raza, A. Poole, M. Güneş, and B. Dezfouli, "Low-power wireless for the Internet of Things: Standards and Applications," *IEEE Access*, vol. 6, pp. 67893–67926, 2018.

[23] P. Gaj, S. Scanzio, and L. Wisniewski, "Guest editorial: Heterogeneous industrial networks of the current and next-generation factories," *IEEE Trans. Ind. Informat.*, vol. 16, no. 8, pp. 5539–5542, Aug. 2020.

[24] P. Thubert, *An Architecture for IPv6 Over the TSCH Mode of IEEE 802.15.4*, draft-ietf-6tisch-architecture-28, 2019, pp. 1–68.

[25] X. Vilajosana, T. Watteyne, T. Chang, M. Vučinić, S. Duquennoy, and P. Thubert, "IETF 6TiSCH: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 595–615, 1st Quart., 2020.

[26] X. Vilajosana, T. Watteyne, M. Vučinić, T. Chang, and K. S. J. Pister, "6TiSCH: Industrial performance for IPv6 Internet-of-Things networks," *Proc. IEEE*, vol. 107, no. 6, pp. 1153–1165, Jun. 2019.

[27] M. Vučinić, T. Chang, B. Škrbić, E. Koćan, M. Pejanović-Djurišić, and T. Watteyne, "Key performance indicators of the reference 6TiSCH implementation in Internet-of-Things scenarios," *IEEE Access*, vol. 8, pp. 79147–79157, 2020.

[28] *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC Sublayer*, IEEE Standard 802.15.4e-2012 (Amendment to IEEE Standard 802.15.4-2011), 2012, pp. 1–225.

[29] D. Stanislowski, X. Vilajosana, Q. Wang, T. Watteyne, and K. S. J. Pister, "Adaptive Synchronization in IEEE802.15.4e Networks," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 795–802, Feb. 2014.

[30] M. Mongelli and S. Scanzio, "A neural approach to synchronization in wireless networks with heterogeneous sources of noise," *Ad Hoc Netw.*, vol. 49, pp. 1–16, Oct. 2016.

[31] M. R. Palattella, P. Thubert, X. Vilajosana, T. Watteyne, Q. Wang, and T. Engel, *6TiSCH Wireless Industrial Networks: Determinism Meets IPv6*. Cham, Switzerland: Springer, 2014, pp. 111–141.

[32] N. Taheri Javan, M. Sabaei, and V. Hakami, "IEEE 802.15.4.e TSCH-based scheduling for throughput optimization: A combinatorial multi-armed bandit approach," *IEEE Sensors J.*, vol. 20, no. 1, pp. 525–537, Jan. 2020.

[33] A. Elsts, S. Kim, H. Kim, and C. Kim, "An empirical survey of autonomous scheduling methods for TSCH," *IEEE Access*, vol. 8, pp. 67147–67165, 2020.

[34] A. Karaagac, I. Moerman, and J. Hoebeke, "Hybrid schedule management in 6TiSCH networks: The coexistence of determinism and flexibility," *IEEE Access*, vol. 6, pp. 33941–33952, 2018.

[35] L. L. Bello, A. Lombardo, S. Milardo, G. Patti, and M. Reno, "Software-defined networking for dynamic control of mobile industrial wireless sensor networks," in *Proc. IEEE 23rd Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, vol. 1, Sep. 2018, pp. 290–296.

[36] M. R. Palattella, T. Watteyne, Q. Wang, K. Muraoka, N. Accettura, D. Dujovne, L. A. Grieco, and T. Engel, "On-the-fly bandwidth reservation for 6TiSCH wireless industrial networks," *IEEE Sensors J.*, vol. 16, no. 2, pp. 550–560, Jan. 2016.

[37] M. R. Palattella, N. Accettura, L. A. Grieco, G. Boggia, M. Dohler, and T. Engel, "On optimal scheduling in duty-cycled industrial IoT applications using IEEE802.15.4e TSCH," *IEEE Sensors J.*, vol. 13, no. 10, pp. 3655–3666, Oct. 2013.

[38] P. Luong, T. M. Nguyen, and L. B. Le, "Throughput analysis for coexisting IEEE 802.15.4 and 802.11 networks under unsaturated traffic," *EURASIP J. Wireless Commun. Netw.*, vol. 2016, no. 1, p. 127, May 2016.

[39] S. Y. Shin, "Throughput analysis of IEEE 802.15.4 network under IEEE 802.11 network interference," *AEU - Int. J. Electron. Commun.*, vol. 67, no. 8, pp. 686–689, Aug. 2013.

[40] C. Ouanteur, L. Bouallouche-Medjkoune, and D. Aïssani, "An enhanced analytical model and performance evaluation of the IEEE 802.15.4e TSCH CA," *Wireless Pers. Commun.*, vol. 96, no. 1, pp. 1355–1376, Sep. 2017.

[41] G. Cena, C. G. Demartini, M. Ghazi Vakili, S. Scanzio, A. Valenzano, and C. Zunino, "Evaluating and modeling IEEE 802.15.4 TSCH resilience against Wi-Fi interference in new-generation highly-dependable wireless sensor networks," *Ad Hoc Netw.*, vol. 106, Sep. 2020, Art. no. 102199.

[42] Q. Wang, X. Vilajosana, and T. Watteyne, *6TiSCH Operation Sublayer (6top) Protocol (6P)*, document IETF RFC 8480, Nov. 2018, pp. 1–50.

[43] P. Thubert, T. Winter, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, *RPL: IPv6 Routing Protocol for Low power and Lossy Networks*, document IETF RFC 6550, Mar. 2012, pp. 1–157.

[44] G. Cena, S. Scanzio, A. Valenzano, and C. Zunino, "A full-wireless network architecture based on the industrial Internet of Things paradigm," in *Proc. 24th IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2019, pp. 1301–1304.

[45] R. Tavakoli, M. Nabi, T. Basten, and K. Goossens, "Topology management and TSCH scheduling for low-latency convergecast in in-vehicle WSNs," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 1082–1093, Feb. 2019.

[46] Z. Shelby, K. Hartke, and C. Bormann, *The Constrained Application Protocol (CoAP)*, document IETF RFC 7252, Jun. 2014, pp. 1–111.

[47] G. Cena, S. Scanzio, L. Seno, A. Valenzano, and C. Zunino, "Energy-efficient link capacity overprovisioning in time slotted channel hopping networks," in *Proc. 16th IEEE Int. Conf. Factory Commun. Syst. (WFCS)*, Apr. 2020, pp. 1–8.

[48] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. 2nd Int. Conf. Embedded networked sensor Syst. (SenSys)*, New York, NY, USA, 2004, pp. 95–107.

[49] L. Seno, G. Cena, S. Scanzio, A. Valenzano, and C. Zunino, "Enhancing communication determinism in Wi-Fi networks for soft real-time industrial applications," *IEEE Trans. Ind. Informat.*, vol. 13, no. 2, pp. 866–876, Apr. 2017.

[50] G. Cena, S. Scanzio, and A. Valenzano, "SDMAC: A software-defined MAC for Wi-Fi to ease implementation of soft real-time applications," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3143–3154, Jun. 2019.

[51] *OpenMote*. Accessed: Jul. 22, 2020. [Online]. Available: https://www.industrialshields.com/shop/product/is-omb-001-open-mote-b-721

[52] *OpenWSN*. Accessed: Jul. 22, 2020. [Online]. Available: https://openwsn.atlassian.net/wiki/

[53] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. J. Pister, "A realistic energy consumption model for TSCH networks," *IEEE Sensors J.*, vol. 14, no. 2, pp. 482–489, Feb. 2014.

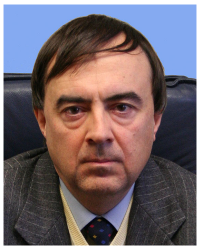
STEFANO SCANZIO (Member, IEEE) received the Laurea and Ph.D. degrees in computer science from the Politecnico di Torino, Turin, Italy, in 2004 and 2008, respectively. From 2004 to 2009, he was with the Department of Computer Engineering, Politecnico di Torino, where he was involved in research on speech recognition and, in particular, has been active in classification methods and algorithms. Since 2009, he has been with the National Research Council of Italy, where he is currently a tenured Researcher with the Institute of Electronics, Computer, and Telecommunication Engineering (CNR-IEIIT), Turin. He teaches several courses on computer science at the Politecnico di Torino. He has authored or coauthored more than 60 papers in international journals and conferences, in the areas of industrial communication systems, real-time networks, wireless networks, the Internet of Things, and clock synchronization protocols. He received the 2017 Best Paper Award of the IEEE Transactions on Industrial Informatics and four Best Paper Awards in IEEE conferences. He served as the TCP Co-Chair of the 2019 edition of the International Conference on Ad Hoc Networks and Wireless and as the Work-in-Progress Co-Chair of the 2018 edition of the IEEE International Workshop on Factory Communication Systems. He is an Associate Editor of the *Ad Hoc Networks* (Elsevier) journal.

MOHAMMAD GHAZI VAKILI received the B.Sc. degree in telecommunication engineering and the M.Sc. degree in mechatronic engineering from the Politecnico di Torino, Italy, in 2017, where he is currently pursuing the Ph.D. degree with the Department of Control and Computer Engineering. His research interests include industry 4.0 (future of factories) focuses on automation, industrial networks, and quantum computing in industry 4.0.

GIANLUCA CENA (Senior Member, IEEE) received the M.S. degree in electronic engineering and the Ph.D. degree in information and system engineering from the Politecnico di Torino, Italy, in 1991 and 1996, respectively. Since 2005, he has been a Director of Research with the Institute of Electronics, Computer, and Telecommunication Engineering, National Research Council of Italy (CNR-IEIIT). His research interests include wired and wireless industrial communication systems, real-time protocols, and automotive networks. In these areas, he has coauthored about 150 technical articles and one international patent. He received the Best Paper Award of the IEEE Transactions on Industrial Informatics in 2017 and of the IEEE Workshop on Factory Communication Systems in 2004, 2010, 2017, 2019, and 2020. He has served as the Program Co-Chairman of the IEEE Workshop on Factory Communication Systems in 2006 and 2008. Since 2009, he has been an Associate Editor of the IEEE Transactions on Industrial Informatics.

CLAUDIO GIOVANNI DEMARTINI (Senior Member, IEEE) received the Ph.D. degree in information and systems engineering in 1987. His teaching activities include graduate-level courses on information systems and innovation management and product development. From 2003 to 2012, he was the Deputy Dean of the Industrial Engineering and Management Faculty and the Deputy Chancellor of the Politecnico di Torino. He was in charge as a member of the Academic Senate and the Head of the Department of Computer Engineering, until 2019. As a member of the Board of Directors of the Education Foundation–Compagnia di San Paolo, he addressed issues related to the Education Community. His main research interests include distributed systems, formal description techniques, software engineering, education, product life cycle, and innovation management.

BARTOLOMEO MONTRUCCHIO (Member, IEEE) received the M.S. degree in electronic engineering and the Ph.D. degree in computer engineering from the Politecnico di Torino, Turin, Italy, in 1998 and 2002, respectively. He is currently an Associate Professor of computer engineering with the Dipartimento di Automatica e Informatica, Politecnico di Torino. His current research interests include image analysis and synthesis techniques, scientific visualization, sensor networks, RFIDs, and quantum computing.



ADRIANO VALENZANO (Senior Member, IEEE) received the Laurea degree *(magna cum laude)* in electronic engineering from the Politecnico di Torino, Turin, Italy, in 1980. He is currently a Director of Research with the National Research Council of Italy (CNR), since 1991. He is also with the Institute of Electronics, Computer, and Telecommunication Engineering (IEIIT), Turin, where he is also responsible for research concerning distributed computer systems, local area networks, and communication protocols. He has coauthored approximately 200 refereed journal and conference papers in the area of computer engineering. He was a recipient of the 2013 IEEE IES and the ABB Lifetime Contribution to Factory Automation Award. He was also awarded for the Best Paper published in the IEEE Transactions on Industrial Informatics in 2016 and the Best Paper Awards for the papers presented at the 5th, 8th, 13th, and 15th IEEE Workshops on Factory Communication Systems (WFCS 2004, WFCS 2010, WFCS 2017, and WFCS 2019). He has served as a Technical Referee for several international journals and conferences, also taking part in the program committees of international events of primary importance. Since 2007, he has been serving as an Associate Editor for the IEEE Transactions on Industrial Informatics.

CLAUDIO ZUNINO (Member, IEEE) received the degree in computer engineering and the Ph.D. degree in software engineering from the Politecnico di Torino, Turin, Italy, in 2000 and 2005, respectively. He has been a Researcher with the Institute of Electronics and Computer and Telecommunications, National Research Council of Italy (CNR-IEIIT), Padua, Italy, since 2006. He has authored or coauthored several papers in international journals and conferences in the areas of wireless communication, industrial Ethernet protocols, computer graphics, parallel and distributed computing, and scientific visualization. He serves as a Reviewer for several international conferences and journals. He has also taken part in the program and organizing committees of many international conferences in the areas of industrial informatics and automation. He has been a Co-Guest Editor of the Special Section on Industrial Communication Technologies and Systems published in the IEEE Transactions on Industrial Informatics.