IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 3, 2019, accepted April 15, 2019, date of publication April 29, 2019, date of current version May 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913703

Modeling and Comparison of Delay and Energy Cost of IoT Data Transfers

SILVIA KRUG AND MATTIAS O'NILS
Department of Electronics Design, Mid Sweden University, 85170 Sundsvall, Sweden
Corresponding author: Silvia Krug (silvia.krug@miun.se)

ABSTRACT Communication is often considered as the most costly component of a wireless sensor node. As a result, a variety of technologies and protocols aim to reduce the energy consumption for the communication especially in the Internet of Things context. In order to select the best suitable technology for a given use case, a tool that allows the comparison of these options is needed. The goal of this paper is to introduce a new modular modeling framework that enables a comparison of various technologies based on analytical calculations. We chose to model the cost for a single data transfer of arbitrary application data amounts in order to provide flexibility regarding the data amount and traffic patterns. The modeling approach covers the stack traversal of application data and thus in comparison to other approaches includes the required protocol overhead directly. By applying our models to different data amounts, we are able to show tradeoffs between various technologies and enable comparisons for different scenarios. In addition, our results reveal the impact of design decisions that can help to identify future development challenges.

INDEX TERMS Analytical models, communication networks, data transfer, Internet of Things, performance evaluation.

I. INTRODUCTION

With a growing interest in various smart systems that benefit from sensing or actuating devices being accessible via the Internet, many different networking technologies to connect the sensor nodes to the Internet have been developed and are currently under development. A large variety of technologies each designed with specific use cases in mind, makes it difficult to select the best solution for a given application or system [1]. Depending on the application requirements, one or multiple technologies can be possible candidates for the last mile communication. Last mile communication in this context refers to the wireless communication within a network from nodes in the field towards a gateway node that forwards the information into a wired network towards the Internet.

Besides criteria such as range and data rate, the cost of the transmission in terms of latency and energy units required to transfer the data amount are important factors. Transmission latency is directly coupled to the operation principles of the technology in question. To transfer arbitrary amounts of application data, it might be required to send several smaller chunks and thus adding to the overall delay. Low energy consumption is a traditional design goal for communication in the Internet of Things (IoT) in order to achieve long node lifetimes. However, often this is achieved by limiting or reducing the data amount to transfer as well as increasing the nodes duty-cycle an thus allowing it to sleep most of the time. If the application requirements are different as for example in [2] and [3], these fundamental assumptions might not hold anymore rendering a technology as inappropriate for the use case.

Therefore, it is important to evaluate the technology-specific cost for each data transfer in order to enable system designers to choose the best option for their constraints. Besides this engineering oriented point of view, such an evaluation also enables a better evaluation of protocol energy efficiency.

Several papers already discuss the suitability of different IoT communication technologies with respect to various application scenarios, e. g. [4]–[8]. However, in those application specific works, the discussions usually focus on a subset of possible technologies and selected use cases covering the traditional periodic traffic pattern only. As a result, a discussion of trade-offs between different technologies is limited. Likewise, the respective applicability of technologies to use cases with requirements that differ from the traditional assumptions remains open.

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

Besides that, there are few tools that allow the evaluation of various IoT communication technologies with different data amounts or traffic patterns. Simulations come to mind as the first option as they provide realistic networking conditions and energy models that are associated to the protocol operation. However, tools like OMNeT ++ [9] or ns-3 [10] only provide a subset of the relevant technologies. Especially recently developed options are missing and thus limiting possible comparisons.

Analytical models are another option to evaluate both cost metrics and several models for the various technologies have been discussed. These models are however technology-specific and are typically developed by different groups with a variety target evaluation goals. Due to this variety, a comparison of the technologies based on multiple separately designed models becomes challenging to impossible, if the parameters and assumptions do not allow a fine tuning of the evaluation scenario. Therefore, a unified framework to evaluate various technologies based on the same assumptions is needed for a fair comparison.

One such approach was presented in [11], where the authors model several low-rate technologies. Other technologies e. g. based on cellular communication are however not covered. Besides that, the focus is to estimate the node lifetime based on periodic traffic. While periodic traffic is common for many applications, it is not the only traffic pattern possible in the IoT context. To reduce data transfers, event-based communication based on in-node processing could also be an option. We contribute to this by building analytical models allowing the evaluation of other traffic patterns as well.

In this paper, we present a framework to analytically estimate the delay to complete the transfer of an arbitrary application data amount as well as the energy required for this. To do this, we follow an analytical modeling approach similar to that presented in [11]. Besides the higher freedom traffic patterns, we extend their work with further possible IoT networking technologies adding especially cellular networks. Realistic energy consumption is achieved by considering both protocol specific operation and real hardware for the energy consumption.

To cover a wide set of technologies, we build the individual models based on a generic description of a single data transfer. Modeling a single data transfer enables the analysis of various traffic patterns or retransmission schemes. Based on the generic description, we classify different technologies based on their relevant operation principles. The individual models are then parameterized accordingly with technology-specific variations where needed. This concept allows an easy extension of the framework with further models, regarding additional technologies or specific Medium Access Control (MAC) protocols.

Using our models, we performed a trade-off analysis regarding arbitrary data amounts and the resulting energy and delay cost metrics. The resulting comparison of available technologies shows some rather surprising effects that are important for application designers and system developers, in order to develop systems that are well suited for the use case at hand. Besides that, the results indicate where further development is needed to enable the full potential of versatile IoT applications, especially if other traffic patterns are applied.

The contributions of our work compared to the state of the art are the following:

- We provide a modular analytical modeling concept for a wide range of IoT communication technologies that is easy to extend with further technologies and hardware.
- We extend the existing models in [11] in several aspects. We add several additional technologies (cellular, 802.15.4g, and multiple Bluetooth Low Energy (BLE) versions). We use real hardware specifications instead of optimized average values and we support event-based traffic patterns in addition to the more traditional periodic traffic.
- Using our framework, we provide results studying the energetic cost of higher data amounts and provide insights in trade-offs between delay and energy cost metrics per technology. In addition, we show that our framework is providing results regarding comparisons of protocol operation, of different hardware options, and node lifetime.

The paper is organized as follows. Before going into the details of current communication technologies for IoT applications in Section III, we briefly introduce the description of a distributed IoT system in Section II that will be used as base for the modeling. Afterwards, we review relevant communication technologies to realize the communication of the actual sensor nodes and describe the relevant criteria for the later modeling. The modeling concept as well as the implementation are described in Section IV. Before comparing the modeled technologies in Section VI, we provide information on performed model validation in Section V. Finally, the paper is concluded in Section VII.

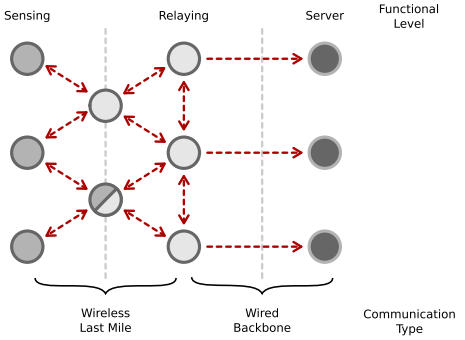


FIGURE 1. Network structure in IoT context with functional levels.

II. SYSTEM DESCRIPTION

An IoT system consists of several functional levels and we deal with different node types at each level as illustrated in Fig. 1 where each level is represented by several nodes. In this paper, we focus on nodes at the sensing level only.

Sensing nodes are nodes deployed at the network edge performing the actual sensing task or, if appropriately equipped, actuation tasks. Depending on the deployment and communication technology, nodes of this type can act as mere data producers/consumers without taking an active part in routing or forwarding information to other nodes or represent a hybrid form between sensor node and relay performing both tasks.

In case of battery-powered or otherwise energy-constraint devices, the nodes energy consumption needs to be well understood in order to achieve guaranteed system lifetimes. Besides the actual sensing hardware, the processing effort and the communication activity are the main consumers with communication being one of the more costly components [12]. The total energy consumption of the node is therefore defined as shown in Equation (1):

$$E_n = E_s + E_p + E_c \tag{1}$$

where E_c in Equation (1) is the energy spend for communication tasks, E_s is the energy required for sensing, and E_p is the energy required for processing tasks.

The energy required for communication tasks (E_c) is depending on the chosen hardware and how often that hardware is used. The activity of the transceiver in turn depends on the application data amount, the protocol stack configuration, and the protocol-specific timing. In this paper, we enable the estimation of both the energy and the delay of a communication task by considering all above points.

Several options exist if communication costs in term of consumed energy are to be reduced: One is to change to a less energy hungry technology with similar performance regarding required data rates. At this point, our models are designed to enable a corresponding comparison between a range of candidates.

Another option can be to change the communication hardware to a more energy efficient version. Whether the impact of this is sufficient, is another question that can be answered by applying our models.

Finally, the amount of data to transfer can be reduced. In this case, the communication effort will be reduced due to less data and thus activity. This does however not guarantee that the overall node energy consumption is lower because some form of in-node processing of the raw data will be required to reduce the data [13]. Since we focus on modeling the communication aspects only, a direct analysis of these in-node trade-offs is not performed. Our work does however support such analyses by enabling the cost estimation for arbitrary data amounts.

III. TECHNOLOGY REVIEW AND SELECTION

Before modeling communication behavior of IoT sensing devices, we will briefly discuss and introduce potential networking technologies for different use cases. We also include an overview on further technology-specific constraints that can limit the suitability of a technology for certain use cases for further comparison and consideration when selecting an appropriate technology. Neither the discussion nor the overview will cover all networking technologies, that have been discussed in the IoT context. In this contribution, we focus on technologies for which corresponding transceiver chips are available and thus allow us to integrate more realistic power values based on the data sheets.

A. COMPARISON CRITERIA

Since we want to model a wide variety of communication technologies covering the typically available options for IoT application designers, the selection has to cover a large set of technologies. We focus on wireless communication technologies that were developed over the last years and where hardware is commercially available. Besides that, additional criteria are important to select an appropriate technology for a given use case. Before going into the details of individual technologies, we therefore introduce the most relevant criteria.

Essential characteristics to evaluate the suitability of a technology for IoT scenarios are the following:

Deployment Options are mainly defined by possible communication ranges and topology options.

Delay is mainly defined by four parameters: the data rate, the frame size, the access probability to the medium and the distance between nodes.

Fairness is mainly depending on the applied access procedure and can have a significant impact on the delay.

Scalability describes the ability of the network to handle traffic from many devices and options to increase the network if needed.

Monetary Cost depends on the required investment in own infrastructure and whether a license fee has to be paid in order to access network resources from other providers.

IP Support is important if all devices are to be accessible from the Internet as well.

At a first glance, the data rate and possible communication range seem to be the most important factors when selecting a technology. Both criteria are highly linked to the respective physical layer (PHY) implementation of a technology. Limiting the comparison to these two parameters only is however too simple.

With respect to *deployment*, there are mainly two options: ad hoc and infrastructure-based communication including cellular networks. In case of ad hoc networks, no previous infrastructure is required and all nodes can talk to each other in a fully meshed topology. This provides flexible deployment options but also the need to deploy the complete system including the relaying layer. Infrastructure-based networks feature fixed gateways that control the communication in their surrounding resulting in star topologies. Here, the nodes cause interference to each other but all communication is mainly handled via the gateway. Only recent enhancements allow direct device-to-device communication of neighboring nodes.

Communication *delay* is another important factor, especially if real-time constraints apply. To achieve real-time

constraints, some form of guarantee is needed for the transmission. Not all technologies provide this, which can result in varying delays for different packets. Besides the technology choice this also depends on the actual deployment and the network conditions in the surrounding of the node in question. Therefore, the delay for a successful transfer will also depend on the actual probability of a node to access the channel when needed as well as the probability for error-free transmissions. Since we focus on the delay to send the data out, the actual transmission delay between two nodes that is affected by the distance between these nodes, is not considered.

Ideally, all nodes should have the same *fair chance to access the medium* within a certain time frame. If such a fair access scheme is ensured, the delay for the medium access should be therefore evenly distributed in average. To ensure this, different medium access schemes are possible. This also applies to scheduled communication, where an entity defines the schedule for all participants.

Scalability is coupled to both the deployment options and the medium access/fairness. If the channel capacity is used up and additional nodes cannot join the network, deploying further resources using different channels is required.

When building an IoT-based system, *monetary cost* is an essential factor. This applies to costs for the nodes at each system level as an initial investment but also to running cost for the operation of the system. Especially, subscription fees required for mobile communication networks are relevant for the latter.

Finally, the *support of Internet Portocol (IP)-based communication* is important, if all nodes in the network shall be accessible via the Internet directly. Most technologies will require some form of gateway to translate the technology-specific formats. If native IP communication is used in the last-mile context as well, further translations/conversion are however not required.

In the next section, we will discuss a wide range of technologies with respect to these criteria.

B. TECHNOLOGY REVIEW

Following the discussion in Section II regarding possible devices and communication technologies for last mile wireless communication, we discuss our selection of technologies here and attempt to cover relevant candidates from different categories. Table 1 summarizes important characteristics of the discussed technologies.

Near Field Communication (NFC) [14] and Radio Frequency Identification (RFID) are two technologies that are mentioned in the IoT context to track and identify objects [15], [16]. Recent RFID tags also feature sensing tasks and can transfer reported sensing data to a gateway [17]. All communication is done in a point-2-point fashion with an extremely short communication range. For NFC a 6Lo-based adaptation is available to enable native IP-based communication [18]. The actual data amount for both technologies is however limited by memory available on the devices, even if the bit rates are also suitable for higher data amounts. Therefore, both technologies are not arbitrarily applicable to different scenarios and will not be considered in the modeling.

The IEEE standardized 802.15.4 family [19] is traditionally applied to Wireless Sensor Networks (WSNs). Different extensions were developed for special applications and multiple PHY variants were introduced. We focus on three variants in this work. The first is the original specification in non-beacon mode and therefore featuring Carrier Sense Multiple Access (CSMA) as medium access scheme. This variant is common for WSN deployments and provides a communication range of around 100 m [20]. As second variant, we chose Time Synchronized Channel Hopping (TSCH) medium access scheme introduced in the 802.15.4 e extension [21] for increased predictability especially in industrial environments [22]. Finally, we consider the sub- GHz PHY introduced in 802.15.4 g [23] with higher data rates and increased range. In addition to the variants standardized by the IEEE, several specialized network stacks can be running on-top of 802.15.4, including ZigBee, Z-Wave as well as Thread.

Bluetooth Low Energy (BLE) [24] is the other frequently discussed technology for short range sensor networks. Besides that, it is a popular technology available in smartphones and used to couple various devices with each other. Typically, BLE provides a star topology with one master device and several slaves connected to it. The master controls the resources assigned to slaves and their sending scheme. We chose three versions that mainly differ in the available data rate but also the support for true mesh communication. The latter was introduced with version 5 [25]. All BLE versions support higher rates than 802.15.4 making them interesting for data intensive applications. IPv6 over BLE [26] is possible and further enables the usage of BLE for IoT applications by enabling IP-based upper layer communication directly without a gateway or translation entity in-between.

In case of industrial applications more standards are relevant to connect the actual devices at the sensing level [27], [28]. One is WirelessHART that was developed as wireless alternative to an existing wired field bus solution HART. Another one is ISA100.11a with basically the same application spectrum and purpose. Both are based on a modified version of the original 802.15.4 PHY with slight modifications and own implementations on the upper layers and especially no direct IP support. The integration into the factory automation network landscape does however limit the applicability of the technologies to industrial contexts only. Similar to the limitations for NFC/RFID, this is the reason why these technologies are not considered further.

If high data rates are required, the more traditional WSN technologies might not be ideal. To compensate this, we include two versions from the WiFi family of standards. The first is 802.11 n [29] which is the current standard used in Local Area Network deployments for example at home or in office environments. Due to this, the infrastructure in form

TABLE 1. Overview on selected IoT networking technologies.

Technology		Category	Type	Topology	Multi-Hop	Frequency	ISM	Bit Rate Downlink	Bit Rate Uplink	Range	Medium Access	Scheme	Frame Size	IP Support
NFC		ad hoc	PAN	point-2-point	no	13.56 MHz	yes	< 400 kbit/s		< 10 cm	slotted ALOHA	message	254 Byte	6Lo
RFID		ad hoc	PAN	point-2-point	no	860–960 MHz	yes	128 kbit/s	640 kbit/s	< 3 m	framed ALOHA		254 Byte	gateway
802.15.4	e	ad hoc	LAN	mesh	yes	2.4 GHz	yes	250 kbit/s		100 m	CSMA	packet	127 Byte	6Lo
	g					sub-GHz		800 kbit/s		800 m	TSCH		2047 Byte	
BLE	4.0 / 4.1	ad hoc	LAN	mesh	(yes)	2.4 GHz	yes	1 Mbit/s		100 m	CSMA	packet	27 Byte	6Lo
	4.2				yes			2 Mbit/s		< 300 m	Master		265 Byte	
	5.0													
WirelessHART		ad hoc	LAN	mesh	yes	2.4 GHz	yes	250 kbit/s		< 100 m	TDMA	packet	127 Byte	gateway
ISA100.11a		ad hoc	LAN	mesh	yes	2.4 GHz	yes	250 kbit/s		< 100 m	CSMA-CA		127 Byte	
802.11	n	ad hoc /	LAN	mesh	yes	2.4 GHz	yes	300 Mbit/s		100 m	CSMA	packet	2304 Byte	IPv6
	ah	infra				sub-GHz		40 Mbit/s		< 1 km	CSMA		2304 Byte	IPv6 / 6Lo
GPRS		cellular	WAN	star	no		no	80 kbit/s	40 kbit/s	> 5 km	Base Station	symbol		(yes)
HSPA								21.1 Mbit/s	5.76 Mbit/s					
LTE	Cat.4							150 Mbit/s	50 Mbit/s					
	Cat.1							10 Mbit/s	5 Mbit/s					
LTE	Cat.M1	cellular	WAN	star	no		no	375 kbit/s		> 5 km	Base Station	symbol		(yes)
	NB-IoT							27.2 kbit/s	62.5 kbit/s					
LoRa		infra	MAN	star	(yes)	sub-GHz	yes	50 kbit/s		< 10 km	ALOHA	packet	222 Byte	gateway
SigFox		infra	MAN	star	no	sub-GHz	yes	600 bit/s	100 bit/s	< 40 km	Random	message	12 Byte	gateway

of access points is usually already available enabling an easy integration of further sensing nodes. Depending on the actual device implementation, 802.11 n supports MIMO (Multiple Input Multiple Output) configurations and data rates up to 300 Mbit/s. These high rate devices are typically not used in embedded systems because of the required energy. Suitable embedded transceiver chips are however available too, at the cost of reduced performance. As second variant of the WiFi, we chose 802.11 ah [30]. This standard was specifically designed for IoT applications providing longer communication ranges and comparatively high data rates [31]. It also supports IPv6 communication via a 6Lo adaptation layer [18].

Cellular communication technologies are also used in the IoT context especially if a long communication range is required. All cellular networks discussed here support packet-based data transfers and IP based communication on top of the lower layer cellular protocols. These technologies are designed with asymmetric data rates favoring the downlink (e. g. from the gateway or base station to the sensing node) instead of the uplink. This scheme efficiently supports short requests from the device and large responses from a server in the Internet as it is found in typical web applications. In the IoT context, this does however not hold as the sensors are mainly producing data and thus require more resources in the uplink. Another critical point is a comparatively high energy consumption to cover the distance to the base stations and the support of parallel users. Recent developments in cellular technologies in 4G and towards 5G leverage these effects.

Among the different cellular technologies, we consider GPRS, HSPA, and the LTE derivates Cat.4, Cat.1, and NB-Iot. Even though it is an old standard, GPRS (General Packet Radio Service) is still relevant, as many sensor applications use GSM-based (Global System for Mobile Communications) modems to send data. Besides that, GPRS is often used as comparison technology for newer options as e. g. in [32]. The same applies to HSPA (High-Speed Packet Access) that provides higher data rates [33]. LTE (Long Term Evolution) supports several device options allowing the users to select appropriate modems for their use case while still using the same network infrastructure. Besides the Cat.4 which corresponds to smartphone class of devices, there are several variants for machine type communication and the IoT. Among these, we selected Cat.1 which features a reduced data rate compared to Cat.4, but has otherwise the same capabilities with respect to voice, Cat.M1 which was specifically designed for machine-to-machine (M2M) communication, and Narrow Band IoT (NB-IoT) designed for low rate, long range IoT communications. The latter two target the typical IoT scenarios with a large number of supported devices that however only send small data amounts infrequently. NB-IoT is part of the LTE specifications (Release 13) but meant to co-exist with LTE or GSM, depending on the operational mode chosen by the maintainer of the infrastructure network [34]. As an LTE sibling, NB-IoT uses the same cellular infrastructure and supports unique PHY features with a focus on uplink transmissions from the sensor devices to the network rather than the traditional downlink direction. Cat.M1 was specified in the same release as NB-IoT and is integrated into LTE bands. In contrast to NB-IoT, it features higher PHY data rates in a symmetric distribution between up- and downlink. Cat.M1 was not modeled for now but might be added later.

Besides these mobile communication standards and the traditional ad-hoc or local area network technologies, several new technologies emerged targeting low rate, long range communications. Among those, we consider LoRa and SigFox. Both technologies target low power wide area networks (LP-WANs) with long communication ranges and represent the recent development towards infrastructure based IoT technologies besides the traditional cellular networks [35]. Another similarity of the two technologies is that they support only comparatively low data rates and long duty cycles. The transmissions typically feature a few bytes of data only. Due to this, both technologies apply custom protocols at the upper layer and are to our knowledge currently not able to support IP-based communication. IP-based communication is only supported between the gateway and other backend servers [35].

Even if the low data rates are similarly limiting as for RFID and NFC, we add LoRa and SigFox to our modeling framework. Both are nonetheless interesting for multiple use cases and provide more interesting networking features thanks to the increased communication range. Besides that, there are attempts to use these networks also for even more challenging applications. One example is described in [36], where the authors verify the use of LoRa for multimedia applications.

IV. ANALYTICAL IOT COMMUNICATION MODELS

In this section, we introduce our evaluation framework that allows to compare the performance of a wide range of last-mile communication technologies.

A. MODELING CONCEPT

To model the communication specific behavior of nodes at the sensing level, requires detailed knowledge on the communication technology as well as the application requirements and the environment of the node. If all aspects are to be considered for different technologies, the resulting calculations become complex. However, it is possible to simplify this considering the goal of our work.

We target models that allow us to estimate the required communication introduced delay and energy consumption of nodes in the sensing level for arbitrary application scenarios. This constraint allows us to reduce the complex network structure to a two-node-system, consisting of the sensor node under test and a receiving counterpart. Two nodes are sufficient to model the ideal communication behavior required to send out data and confirm that transfer on medium access level. Therefore, we chose to model the chosen communication technologies analytically.

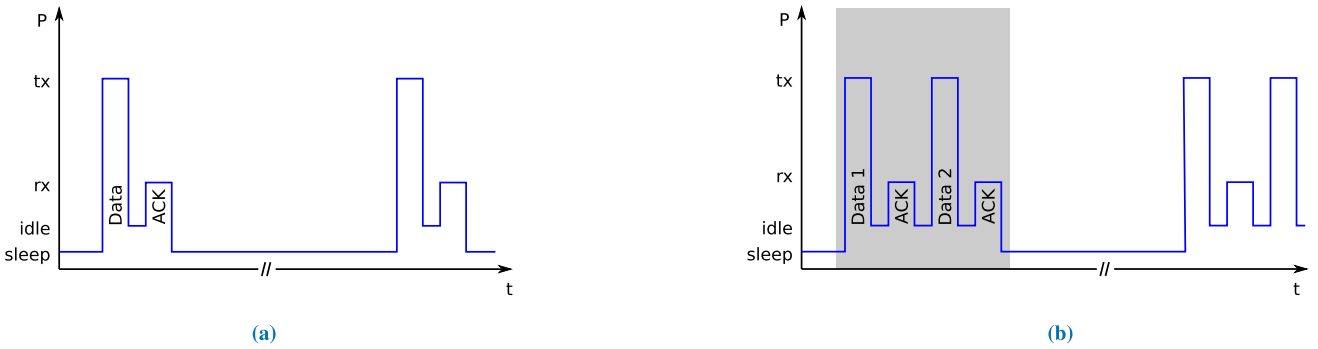


FIGURE 2. Different Duty-cycling schemes. (a) Traditional. (b) Multi-transmission.

The required level of detail for the two nodes is however different. On the sender side, the delay to send a specific amount of data is depending on the factors mentioned in Section II and is therefore protocol and application-specific. The same is true for the energy required for the transfer, but the actual power consumption depends additionally on the desired hardware. Therefore, the model of the sensor node sending its data has to be a white box model including details with respect to both the technology in question and transceiver hardware.

On the receiver side, a simpler black box model is sufficient. The model of the receiver has to describe the protocol-specific reaction to received data in order to estimate the correct timing of the interaction between sender and receiver. Further details on transceiver hardware or the role of the receiver in the network, are not required as they do not effect the energy cost at the node under test. The same is true for potential further transfers, that might be required to reach the final destination after the initial reception of the data at the first hop. Fig. 3 depicts this basic system and general modeling idea.

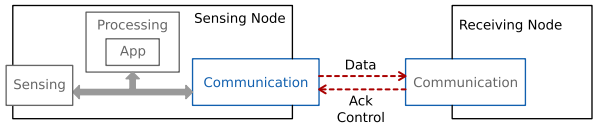


FIGURE 3. Simplified two node system as modeling base.

Reducing the system to two nodes only, leads to a system without interference from other nodes. Interference and interactions between multiple nodes at each functional level have an impact and will result in both increased delay and increased energy consumption. The increase results from a potentially reduced resource availability or repeated transmissions in case of errors. Both aspects lead to a higher transceiver activity than under ideal conditions even if we only consider the sensing node as data source.

Using the two-node-system, the estimated cost metrics will represent the minimum achievable cost regarding both energy and delay and thus the best case for a given scenario. This best case scenario provides us with valuable insights regarding the suitability of a technology for that use case and thus a tool to evaluate different design choices when selecting one or multiple communication schemes for a given application.

The white box model for the sending node follows the approach in [37] and is modeled around two principle equations. Since modern transceiver chips provide several power consumption levels depending on the current operation mode, it is crucial to estimate how much time is spent in each state. This timing is defined by the protocol operation and the amount of data to send. We chose to consider four main states: sending tx, receiving or listening rx, idle i, and sleeping s. The actual transceiver activity when completing one transfer of application data includes the sending tx, the active listening rx, an *idle* state when the transceiver is active but neither sending or listening, and switching times between these states. In addition the transceiver itself typically has several low power or sleep modes, that are used between subsequent transfers. Using these states, we calculate the total time it takes to send out the application data amount (further denoted as delay in this paper) in Equation (2) as the sum of the time spend in each state.

$$t_c(d) = t_{tx}(d) + t_{rx}(d) + t_i(d) + t_s(d) \tag{2}$$

The total energy (cf. Equation (3)) is then calculated accordingly by multiplying the duration spent with the hardware specific power consumption in each state:

$$E_c(d, hw) = t_{tx}(d)P_{tx}(hw) + t_{rx}(d)P_{rx}(hw) + t_i(d)P_i(hw) + t_s(d)P_s(hw) \tag{3}$$

where d denotes the application data amount to send and hw denotes a specific transceiver. Depending on the actual protocol operation, some states are optional. In that case, the corresponding term in Equation (3) equals zero.

Using all four states, we are able to describe event-based and periodic traffic with traditional duty-cycles (cf. Fig. 2a). We extend the duty-cycle concept in order to support arbitrary application data amounts. If the desired amount is too large for a single transmission, we model multiple subsequent transmissions based on the technology operation principles (cf. gray highlight in Fig. 2b). Due to the subsequent transmission, longer waiting phases in between packets can occur.

This duration is modeled with the transceiver in sleep mode. Therefore, we include the term $t_s(d)P_s(hw)$ into the Equations (2) and (3). TSCH is one technology where this is possible due to the scheduling.

The goal of the model is to identify how much time is spent in each state based on a given data amount and chosen technology by estimating the number of transmissions $n_T(d, te)$ at transceiver level. Before going into details about this, we will introduce our assumptions to enable analytical descriptions of the communication.

B. ASSUMPTIONS

Besides the limitation to a two-node model, we took the following additional assumptions:

1) Only data transfer modeled
2) Only single transfer modeled
3) Overhead includes IPv6 headers where applicable
4) Energy consumption based on datasheet values

We model a confirmed data transfer. This means that any additional control traffic required to setup and maintain a connection is omitted from the model. Only those control messages directly related to the data transfer (acknowledgements and resource reservations) are considered. Therefore, the resulting delay represents the time required to send the data and receive a confirmation that it was received successfully.

Our model returns the metrics of one single transfer, only. Transfer here refers to one amount of application data, independent of how many actual physical transmissions are required to transfer that amount. It therefore does not support direct calculations of a nodes lifetime, which is another frequently used metric for the evaluation of IoT technologies. Such a lifetime estimation requires further constraints regarding the traffic patterns, e. g. periodic sending intervals. These constraints however limit the potential of the tool to consider multiple use cases with diverse application requirements. On the other hand, calculations of the lifetime as well as analyses of shortest sending intervals become possible based on the resulting values for delay and energy values.

It is unlikely to send application data as raw data via the communication interface directly. Usually, the firmware of a node contains a network stack implementation supporting a selection of higher layer protocols depending on the nodes memory and processing capabilities. As mentioned in Section III-A, we favor IP-based communication where possible. Therefore, we have to include corresponding header information and possible fragmentation as overhead. This means that the actual data amount transferred by the interface can be significantly higher than the raw application data due to additional control overhead. We consider this fact by integrating the overhead into the model.

In order to estimate realistic energy consumption for the data transmission, we use real transceiver hardware as a base for the energy calculations. All energy consumption values (cf. Tables 3 and 4) for different activity states of a transceiver are taken from the respective datasheet. We selected hardware that is available on the market and used for embedded applications. The latter fact can lead to reduced performance characteristics in order to save energy.

C. MODELING FRAMEWORK OVERVIEW

Several aspects are involved if the required time for an application data transfer has to be estimated, independently of the chosen technology under review. Fig. 4 shows the flow of information through the node.

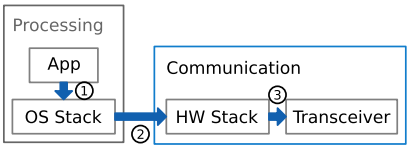


FIGURE 4. Information flow between application and transceiver.

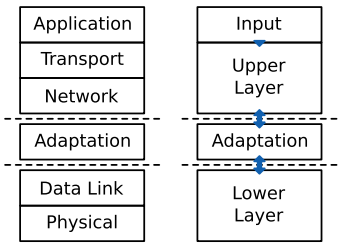


FIGURE 5. OSI model for IoT communication and corresponding model components.

Besides the raw application data (1) and the transceiver hardware, the information has to traverse the network stack of the node. Part of the stack is implemented as operating system (OS) or firmware function while another part is hardware implemented. The latter typically applies to the lower layers of the Open Systems Interconnection (OSI) reference model (cf. Fig. 5).

Even though we only model the communication module aspects here, we have to consider the OS-related network stack functions in order to obtain the correct data amount at the transceiver (2). While traversing the network stack, each protocol layer adds additional encapsulation overhead. To model proper timing of a data transfer, the overhead introduced by the stack configuration has to be considered only at the PHY layer (3), the bit rate and operation principles define the actual duration of transceiver activity.

As a result, our model follows the data flow through the network stack and considers the package encapsulation at each layer. However, we do not model each layer separately. We simplify the models, by grouping relevant functionality instead. Fig. 5 shows this concept.

The grouping is done based on similar behavior of the technologies at each level. For example, all IP-based technologies feature the same upper layer implementation, while the adaptation layer is parameterized based on the chosen lower layer technology. The parameterization is used to describe

the interdependencies of the three components. Using this concept, we are able to describe each individual technology model as a combination of the three main components *Upper Layer*, *Adaptation*, and technology specific *Lower Layer*.

Upper Layer wraps the application data into packets that are then transported by the lower layers.

Adaptation describes the process of adjusting typically large packets from the Upper Layers into smaller transmission units at the Lower Layers. This layer is optional and will be used only if required by the technology in question.

Lower Layer combines the PHY and MAC layers. Implementations of this component model therefore the specific operation principles at the PHY layer and the idealized medium access.

Fig. 6 shows the class diagram of the framework.

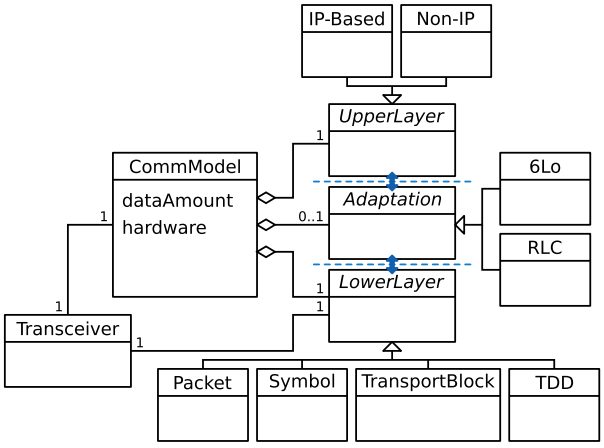


FIGURE 6. Class diagram of modeling framework.

Each of the three main components is designed as interface defining the common behavior of the component. Derived classes implementing the interface are then used to model the unique differences. Table 2 gives an overview of the combinations used to model the individual technologies.

To model the Lower Layer functionality, we derived four abstract classes describing the common operation principles of different technologies. Specific technologies are then modeled by parameterizing and extending the general class.

Packet describes a PHY/MAC combination that is focused on the transmission of data in form of packets or frames. Examples are BLE and traditional 802.15.4.

Symbol describes a combination, that uses packets as input, but then assigns the data to a stream of symbols. Examples are WiFi or 802.15.4 g with Orthogonal Frequency Division Multiplex (OFDM) PHY.

TransportBlock describes the LTE PHY operation where the data is matched to Transport Blocks (TB). Even if the underlaying technology uses one or multiple symbol streams, the general operation principle focuses on transferring one TB size data chunk per transmission interval. This version is used for all technologies based on 3GPP standardization.

TABLE 2. Model configurations.

Technology	Model Component		
	Lower	Adaptation	Upper
802.15.4 CSMA	Packet	6Lo	IP-based
802.15.4 TSCH	Packet	6Lo	IP-based
BLE 4.0 / 4.1	Packet	6Lo	IP-based
BLE 4.2	Packet	6Lo	IP-based
BLE 5	Packet	6Lo	IP-based
SigFox	Packet	none	Non-IP
802.11 n	Symbol	none	IP-based
802.11 ah	Symbol	none	IP-based
802.15.4 g	Symbol	6Lo	IP-based
LoRa	Symbol	none	Non-IP
LTE Cat. 4	TB	RLC	IP-based
LTE Cat. 1	TB	RLC	IP-based
NB-IoT	TB	RLC	IP-based
HSPA	TB	RLC	IP-based
GPRS	TDD	none	IP-based

TDD describes a PHY where the nodes get assigned time slots to transmit their data. This model is chosen for GPRS with is modeled on the Time Division Duplex (TDD) frame structure of GSM.

The design of our modeling framework allows further extension for additional technologies or other components by implementing additional variants for each main component. Additional transceiver hardware can be added similarly by parameterizing a corresponding *transceiver* object.

D. MODEL IMPLEMENTATION

To implement our framework, we developed MatLab functions for each of the classes defined in Fig. 6 featuring the corresponding equations to capture the fundamental operation principles of each technology. For each technology, a wrapper function takes care of the interaction between the components according to Table 2. In all figures of the this section, we use arabic letters from (a) to (k) denote the size of individual packet components such as header, payload or tail. The actual value will be derived from the protocols at hand in each case. The code is available for download under [38].

1) UPPER LAYER

The Upper Layer components handle the encapsulation of the application data and thus define the amount of data to be handled by the adaptation and lower layers. Fig. 7 illustrates this concept for the IP-based version featuring UPD as Transport Protocol. In case of the non-IP version, no additional overhead is added and the fragmentation is done at the lower layer.

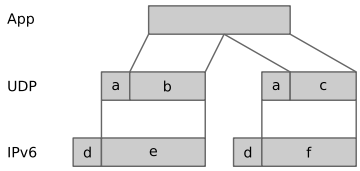


FIGURE 7. Overhead and fragmentation mechanism of IP-based Upper Layer component.

We chose UDP over TCP for the initial version of the model as it is more common for IoT applications due to the reduced overhead. The concept is however not limited to UDP and further implementations can be added easily. Adding TCP is therefore possible and might be implemented in the future.

In Figure 7 (a) denotes the UDP header udp_{he} size and (b) the corresponding payload. We chose the UDP payload size so that one UDP packet fits into the payload of a maximum-sized IP packet (e). The Maximum Transmission Unit (MTU) of an IPv6 packet ip_{mtu} is set as parameter in each model according to the defined standards. It defines the maximum size of a packet including the full IPv6 header ip_{he} (d) information. As a result, the maximum payload of a UDP and IP packet (cf. Equation (4)) is therefore defined based on the MTU and the header information:

$$ip_{pl} = ip_{mtu} - (ip_{he} + udp_{he}) \tag{4}$$

Based on this payload value, the number of full-sized IP packets $n_{ip}(d)$ and an optional final packet with a shorter payload (c and f) is calculated. This information is then provided to the component below.

2) ADAPTATION LAYER

Regarding the Adaptation Layer component, two subclasses are currently implemented: one for 6Lo and one for 3GPP defined PDCP and RLC layer combination as discussed above. In case of the **6Lo model**, all technology-dependent header and payload sizes are provided as input values to the function, allowing flexible configuration. The parameters are taken from the overview on different 6Lo variants provided in [18]. The model itself is built as follows.

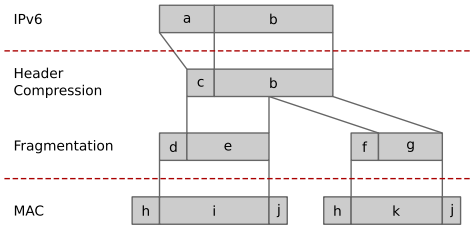


FIGURE 8. Segmentation and compression mechanisms in 6Lo Adaptation layer components.

In traditional sensor networks, IPv6 packets have to be fragmented again in order to fit into comparatively small packets at the MAC level. 6Lo was introduced to do this and is now standardized for multiple underlying technologies [18]. Fig. 8 illustrates the concepts supported by our model. The dashed lines indicate interfaces to the upper and lower layer components.

Based on the IP packet information (size and number of packets) from the Upper Layer component, the number of frames at the MAC level is calculated. 6Lo allows to compress the IP header (a) to by removing all redundant information. As a result the header becomes smaller (c). We modeled this feature as optional, as some implementations do not support full header compression [39]. This allows us to estimate the energy consumption for different real implementations as well.

The next step is the segmentation of the IP packet into smaller fragments. (d,f) denote the individual header $6lo_{he,te}$ of each fragment. The size of the first header can be different than that of subsequent fragments [18]. Since the total fragment size is restricted by the maximum payload size of the MAC frame (i), the fragment payload $f_{pl,te}$ varies as well. Four cases are possible: 1) the first maximum-sized fragment (cf. Equation (5)) , 2) all subsequent maximum-sized fragments (cf. Equation (5)) , 3) the final shorter fragment of each full-sized IP packet $6lo_{pl,l,te}$ (cf. Equation (6)) , and 4) the final fragment of the final shorter IP packet $6lo_{pl,f,te}$ (cf. Equation (7)). This distinction is required to calculate the duration spend in the *tx* state.

These cases have to be handled separately, as they result in differently sized MAC frames. Equations 5 to 7 give the corresponding fragment sizes that can occur.

$$6lo_{pl,m,te} = f_{pl,te} - 6lo_{he,te} \tag{5}$$

$$6lo_{pl,l,te} = ip_{mtu} \bmod f_{pl,te} \tag{6}$$

$$6lo_{pl,f,te} = (d \bmod ip_{mtu}) \bmod f_{pl,te} \tag{7}$$

Besides the size of each fragment type, also the number of fragments per type is required for the timing calculations. The number of full fragments (cf. Equation (10)) is calculated based on the IP MTU and the Lower Layer frame size (cf. Equations (8) and (9)).

$$n_{6lo_m,ip} = \left\lfloor \frac{ip_{mtu}}{6lo_{pl,m,te}} \right\rfloor \tag{8}$$

$$n_{6lo_m,ip_l} = \left\lfloor \frac{d \bmod ip_{pl}}{6lo_{pl,m,te}} \right\rfloor \tag{9}$$

The result of Equation (10) accounts for all maximum sized fragments of IP packets with MTU size and all maximum sized fragments of the final, potentially smaller, IP packet.

$$n_{6lo_m,te}(d) = \left(n_{ip}(d) - 1\right) \cdot n_{6lo_m,ip} + n_{6lo_m,ip_l} \tag{10}$$

Correspondingly, $n_{ip}(d) - 1$ fragments with a size according to Equation (6) and one fragment with the finally remaining data amount according to Equation (7) are required to transfer the complete arbitrary application data amount. The total number of fragments and thus transmissions is therefore calculated as shown in Equation (11):

$$n_{6lo,te}(d) = n_{6lo_m,te}(d) + n_{ip,te}(d) \tag{11}$$

Information on how many fragments of each type exist and what their actual size is, is than handed to the lower layer component to calculate additional MAC/PHY overhead as well as the delay due to medium access.

The **Radio Link Control model** works similar in principle, starting at the Packet Data Convergence Protocol (PDCP) layer in order to obtain the number of Transport Blocks required to transfer the data amount from the upper layer following the data flow described in [33] and [40].

In cellular networks up to the 4 generation, two protocols are used between IP and the actual MAC layers. These protocols are modeled as adaptation layer component with the details shown in Fig. 9.

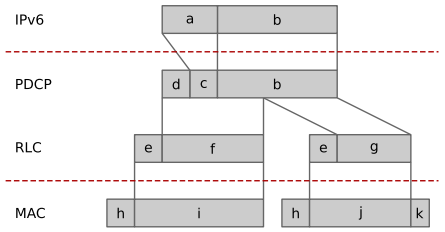


FIGURE 9. Segmentation and compression mechanisms in RLC Adaptation layer components.

When traversing the stack from IP downwards, the packet is first processed by PDCP. This protocol is responsible for header compression as well as several other functions for the connection control [40]. The later functions are not considered here. PDCP reduces the size of the IP header from (a) to (c) and adds its own header (d) to each packet. The payload size (b) remains the same at this point and forms the PDCP packet payload $pdcp_{pl}$ (cf. Equation (12)) together with the compressed header $ip_{he,comp}$.

$$pdcp_{pl} = ip_{pl} + ip_{he,comp} \tag{12}$$

The second protocol is the RLC protocol. It is able to concatenate and fragment PDCP packets in order to provide maximum-sized payload (i) Transport Blocks. To ensure this, padding (k) is used to fill up the final packet, if needed. RLC adds its own header information (e) to each newly generated packet, too.

Similar to the 6Lo Adaptation Layer component, the size and number of the different fragment types has to be calculated. Since the combination of two protocols results in data reduction followed by data expansion, this is done in a two step process. First, we calculate the amount of data after the PDCP processing according to Equation (13) including all headers. Afterwards, we perform the segmentation of that amount at RLC level to obtain the number of MAC frames $n_{rlc,f}(d)$ based on Equation (14).

For the first step, we use the number of full sized IP packets $n_{ip}(d)$ and the size of the final IP packet $ip_{pl,f}$ as provided by the upper layer component:

$$data_{pdcp} = n_{ip}(d) * (pdcp_{he} + pdcp_{pl}) + (pdcp_{he} + ip_{he,comp} + ip_{pl,f}) \tag{13}$$

Each IP packet results in one PDCP packet, with the difference in size for the final IP packet, if that has a smaller size $ip_{pl,f}$.

The second step is based on the maximum size of payload mac_{pl} that can be transported by the lower layer (i). This values defines the fragment size at the RLC layer. Each fragment gets an RLC header rlc_{he} (e). Since padding (k) is used on the MAC layer to fill up any final shorter fragment (g), we can use the ceiling operation in this case to obtain the number of lower layer transmissions $n_{rlc}(d)$.

$$n_{rlc}(d) = \left\lceil \frac{data_{pdcp}}{mac_{pl} - rlc_{he}} \right\rceil \tag{14}$$

Here, we use the accumulated data at the PDCP layer that is required to transfer the given application data amount according to Equation (13) as input value. The amount is then divided by the payload available at RLC in order to obtain the number of transmissions required. This information is then handed to the lower layer component for actual transportation.

3) LOWER LAYER

Each Lower Layer component covers the MAC and PHY layers of the OSI reference model of a chosen technology. These two layers are traditionally covered by the respective standardization documents of that technology. Our modeling approach follows the behavior in the standards for both encapsulation and medium access. Therefore, we require one function per technology.

We did however group the functionality into four general schemes, covering common operation principles. Due to this, the PHY management is quite similar for all representatives of each class except for the technology-dependent parameters, that are modeled as constants at this level. For each version, we also include PHY and MAC overhead per transmission. Therefore, our model covers the total overhead throughout the stack more realistically than for example the models described in [11].

In order to calculate the actual timing and energy consumption of a transceiver, the data amount and the required number of transmissions at the PHY layer are however not enough. What is missing are delays introduced by technology-specific medium access schemes. Therefore, we include specific variants for each technology covering its characteristic medium access as defined in Table 1. The modeling of that follows the standardized access schemes and is not repeated here.

Packet-based as well as symbol-based models use the MAC frame size according to Table 1 as the base unit. In case of **packet-based transmissions**, the lower layer takes the information on size and number of fragments from an level above and uses that to encapsulate the data into MAC frames. For each frame MAC overhead in form of header (a) and tail (c) is added to the payload (b) before it is handed down to the PHY layer, which adds additional overhead (d). Fig. 10 shows this. To obtain the timing information, we calculate the

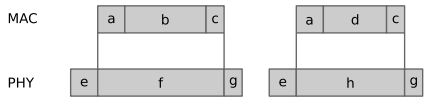


FIGURE 10. Encapsulation of data in packet-based Lower layer components.

number of transmissions for each frame size by considering the PHY data rate including all overhead.

For **symbol-based** transmissions, we calculate the number of symbols required to represent the frame information including overhead and use the symbol duration to evaluate the timing. The data is encapsulated into corresponding MAC frames first containing a header (a) and the actual payload from the upper layer (b and c). Similar to the packet-based approach, a PHY preamble is sent before the actual frame. The preamble (d) and the payload (e,f) are symbol representations. Fig. 11 illustrates this.

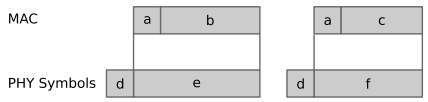


FIGURE 11. Encapsulation of data in symbol-based Lower layer components.

The standard documents contain information on how to calculate the number of symbols for a given data amount at the MAC level. In addition, they provide equations to calculate the transmission time required for a number of data symbols as well as any control packets that might be needed to provide a confirmed transmission service. These equations are specific for each technology.

In some cases, the MAC layer contains several aggregation mechanisms as for example for the WiFi standards. This information is considered in the model as well, adding further intermediate steps to the encapsulation.

The **TDD model** is used to describe GPRS since it follows the timeslotted medium access of GSM. We assume that the sensing node is provided with 5 time slots per GSM frame using two of them for uplink communication. Based on this assumption and the GSM frame structure, we are able to calculate the timing for a given amount of time slots.

To obtain this number, we have to consider the encapsulation between the IP-based Upper Layer and the slot capacity. Fig. 12 shows the encapsulation scheme.

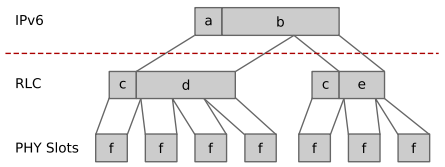


FIGURE 12. Encapsulation of data in TDD-based lower layer component.

The IP packet is fragmented into RLC blocks with a maximum-sized payloads (d) and a potentially shorter final fragment (e). One RLC block including the header (c) is transferred in four slots (f) including all overhead. Even if a packet is shorter than the maximum capacity of the slot, the transceiver stays active for that duration.

More recent cellular technologies are not using the time slot concept but a more flexible resource allocation where a variable length **TB** is transferred during a Transmission Time Interval (TTI) [33]. The size of the TB corresponds to the current resources allocated to the node by the base station and is depending on current channel conditions as well as cell status. In our model with two nodes, we assume a best case scenario resulting in maximum sized TBs per technology for the given node and enough resources for subsequent data packets if required.

In order to obtain the number of TBs, we consider the encapsulation. We use the TB size (e) as base parameter to define the maximum size of a MAC frame, including header (a), payload (b, c) and potential padding (d). Padding is used at the MAC layer to fill up shorter frames. Fig. 13 visualizes the encapsulation scheme.

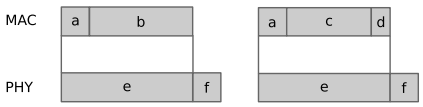


FIGURE 13. Encapsulation of data in lower layer components based on transport blocks.

Each TB is secured with a checksum (f) at the PHY and is transferred according to the resource assignment in a TTI. We calculate the timing for the complete data transmission based on the number of TBs and the TTI duration.

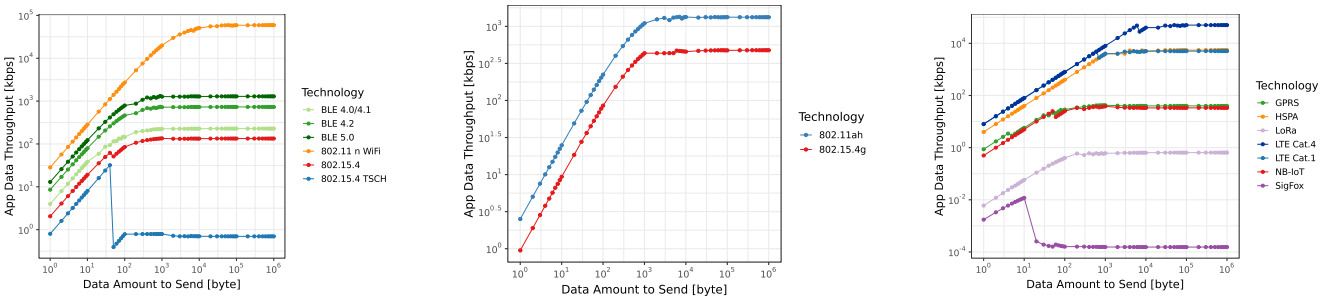
4) TRANSCEIVER MODEL

To obtain a realistic energy consumption, we used voltage and current values reported in the datasheets of transceiver hardware. We chose this approach over measurements to provide a fair comparison between many technologies. It is however possible to replace the datasheet values with actual measurement results for the corresponding activity states. In the following, we introduce the chips and their consumption values that we use throughout the remaining evaluation.

Table 3 lists the transceivers for ad hoc technologies. We chose a sleep mode where the chip is running one timer and uses memory retention to ensure that the nodes can wake up without requiring an external trigger.

Another important point to note is that two transceivers are capable to run both 802.15.4 and BLE. The category in the table corresponds to the technology that we use for the comparisons in Section VI. However, the models support the analysis of both technologies in this case.

Table 4 lists the corresponding parameters for the cellular transceivers. In this case, several transceivers do not distinguish current consumption for transmit and receive modes. Therefore, we only list the combined value when the transceiver is active. The sleep current in this case has the



(a) (b) (c)

FIGURE 14. Throughput (a) Short Range. (b) Medium Range. (c) Long Range.

TABLE 3. Hardware overview ad hoc technologies.

Tech.	Chip	Current RX mA	Current TX mA	Current Sleep[1] µA	Voltage V
802.15.4	CC2520	18.5	25.8	175.0	3.0
	CC2538	20.0	24.0	1.3	3.0
	SAMR21	11.8	11.8	4.1	3.3
	nRF52840[2]	5.9	6.1	1.0	3.0
802.15.4g	AT86RF215	28.0	64.0	0.3	3.0
BLE	CC2540	15.8	21.0	235.0	3.0
	CC2560	6.4	5.3	40.0	3.0
	nRF51822	9.7	8.0	1.2	3.0
	da14585	3.1	3.4	1.4	3.0
	kw41z[1]	6.8	6.1	1.1	3.0
802.11n	ATWINC1500	83.0	287.0	4.0	3.3
802.11ah	PCS HaLow	7.3	12.0	1.0[3]	3.0

[1] Using low power mode with active timer
[2] Transceiver supports 802.15.4 and BLE
[3] Estimated based on [41] due to datasheet unavailability

TABLE 4. Hardware overview cellular technologies.

Tech.	Rel.	Chip	Current RX mA	Current TX mA	Current Sleep mA	Voltage V
3G						
GPRS	–	SARA G3	300		0.9	3.8
HSPA+	7	SARA U2	425		0.9	3.8
LTE-based						
Cat.4	9	TOBY L2	610		2.7	3.8
Cat.1	9	LARA R2	540		1.4	3.8
Cat.NB1	13	SARA N2	46	220	6.0	3.6
IoT-specific						
LoRa	–	SX1276	13.8	28.0	1.5 µA	2.0
SigFox	–	ATA8529D	10.4	31.8	5.0 nA	3.8

same definition as in Table 3 for NB-IoT, LoRa, and SigFox. For the other transceivers this corresponds to the idle mode, where the node is still connected to the network.

Transceivers for the IoT-specific technologies NB-IoT (Cat.NB1), LoRa, and SigFox provide distinct values for each state. Especially the latter two technologies reach low current consumption for all states. The values are however still higher than those of recently developed transceivers for ad-hoc technologies.

V. MODEL VALIDATION

In this section, we describe how we validated our approach. As mentioned before, there are three general options to analyze the performance of a given technology. These can also be used to validate our approach. However, the limitations that lead to the development of this tool have to be considered during validations as well. To justify the approach, we therefore use a combination of options.

A. THROUGHPUT ANALYSIS

As a first evaluation, we estimated the achievable application data throughput of each model. Our model estimates the cost in terms of delay for a single data transfer of a given amount of application data and includes all waiting times and repetitions as required by the technology under test. To calculate the throughput, we divide the input data amount by the delay required for the transfer as calculated by our models. Since we model UDP and IP overhead as well as technologies specific timing constraints, we expect that the calculated throughput stays below the nominal data rate of each technology as described in Table 1. Fig. 14 shows the calculated throughput of a single transfer of application data in the range from 1 Byte to 1 MByte. We chose large amounts to illustrate the convergence of the throughput to the link capacity as well as to illustrate the impact of higher data amounts on the transfer in general. Details on delay and energy costs are discussed in the following section.

For better readability the technologies are grouped based on their average communication range. Based on the values in Table 1, we define all technologies with ranges smaller than 300 m as *short range*, technologies with ranges over 1 km as

long range, and all technologies between these as *medium range*. The observed fluctuations result from fragmentation and consequently different transmission unit sizes. Whenever an additional transmission is required, the overhead increases and therefore the throughput decreases.

As expected, the throughput values stay below the expected throughput of the technologies and show a saturation towards the link capacity. However, this throughput analysis contains some surprising results too. First, the performance of TSCH does not follow the general trend. Instead, it decreases significantly with increased data amounts. The reason for this is the scheduling scheme used. We model TSCH with a default slot frame length of 101 slots of 10 ms and one assigned slot per node [42]. As a result, transfers that require multiple slots, get delayed disproportionally leading to the observed throughput performance.

Second, the performance of SigFox shows a significant drop and stays low afterwards. The drop happens as soon as more than one transmission is required to send the data amount from the application layer. Due to the strict timing regulations for SigFox, each additional packet requires a significant waiting time required for a 1% duty-cycle. This leads to the very low throughput.

B. COMPARISON TO MEASUREMENTS

To evaluate the delay calculations, we performed measurements with different UDP payloads 802.15.4. We used two OpenMotes [43] programmed with an adapted version of the gnrc_example of RIOT [44], [45] in order to send arbitrary application data amounts over UDP and 6LoWPAN. The stack was configured to support a maximum size of 1232 Byte at the 6LoWPAN layer corresponding to one IPv6 packet. Larger data amounts are fragmented manually in the application. The delay was measured in software, by generating timestamps after each sending event.

Using this setup to send and receive data between the two 802.15.4 devices. The results vary by 5 to 8% between measurement and estimation using our model for cases where more than one frame is required to transfer the application data. The difference is bigger for smaller data amounts, where the measurement shows higher values.

C. COMPARISON WITH LITERATURE

Finally, we can compare out results to previously reported performance of the technologies in the literature. This is especially possible for the energy consumption, because many different works target either energy modeling or characterization of various technologies. Most other approaches focus on node and on network lifetimes while we model a single data transfer in order to provide some flexibility regarding possible traffic patterns..

Therefore, some additional calculations are required for a comparison of node lifetime. This includes the number of transfers and the duration spent duty-cycling between subsequent transmission events. Fig. 15 shows such an example comparison using our models to predict the energy

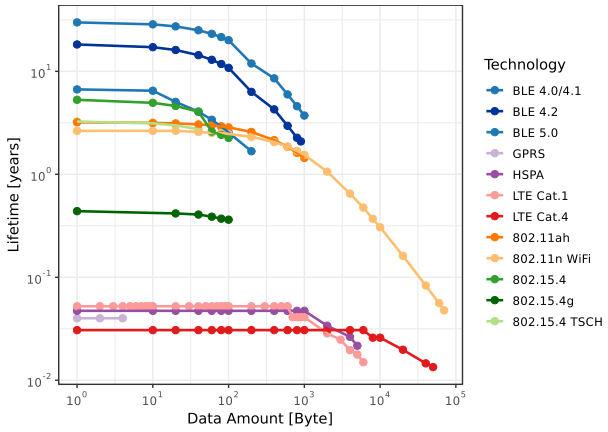


FIGURE 15. Node lifetimes for periodic traffic.

consumption for periodic data transfers. The sending interval is set to 1 s and the data amount is varied. Otherwise, we assume the same energy model and setup as described in [11] featuring an initial energy amount of 13.5 kJ and a cutoff voltage at 10% of that energy.

As a constraint, we only considered technologies that are able to complete a single data transfer within 10 ms. Due to this, LoRa, NB-IoT and SigFox are not part of the comparison and some of the technologies are not able to cover the complete range of possible data amounts. We use the sleep mode (cf. Tables 3 and 4) of each transceiver to model periods of inactivity. This results in an optimistic estimation, because additional transceiver activity might be required to remain connected to the network [12]. Therefore, we expect real world lifetimes to be below our estimation.

When comparing our results to those reported in [11], we can observe that our model reports overall shorter node lifetimes while the general behavior stays the same. This difference results from a different approach regarding the hardware parameters. Morin *et al.* [11] use several actual transceivers and select minimal values for each transceiver state from the list instead of using values from a single chip. Therefore, their lifetime calculations are based on idealized minimal consumption values which result in longer system lifetime. This is however misleading as the variance between different hardware for the same technology is quite high as shown in Table 3. The lifetime results already show that cellular technologies have a high cost, due to energy hungry transceivers (cf. Table 4). More details on how this affects single transfer comparison are provided in the next section.

Apart from the node lifetime, other parameters can be used for comparisons as well. Del Carpio *et al.* [46] report a single-hop delay for 802.15.4 of 5 ms for a payload at MAC layer of 120 Byte. When adapting the application payload by reducing the payload introduced by UDP, IPv6 and 6LoWPAN, our model reports 4.35 ms. In the same work, 802.11ah and BLE 4.2 are compared, both are reported with a delay of approx. 4 ms. Our model returns 3 ms for 802.11ah and 2 ms for BLE. The difference results from our assumption to

consider only the data transfer and not preceding connection setups.

Regarding BLE, Spörk *et al.* [47] provide measurement results for uplink packet duration and energy consumption for a transmission of 27 Byte fragments. Calculating the total delay for a single transfer based on their input parameters and using the fragmentation required for IPv6 over BLE, we are able to reproduce the same results. Morin *et al.* [11] report a delay of 6.42 s for a 12 Byte SigFox frame. The same value is reported by our tool. The performance of LoRa was analyzed in [48]. According to that work, it takes 1.45 s to send a LoRa frame with 20 Byte payload that is sent with SF = 12, which matches exactly the value from our model. For NB-IoT and GPRS the time-on-air per 10 Byte packet is reported in [32]. The time-on-air corresponds to the delay estimated by our model within 5% of the originally reported value.

VI. COMMUNICATION COST TRADE-OFFS

In order to evaluate the cost trade-off for different communication technologies, we performed several comparisons using our tool. The following results present the operation of the tool and show how is can be used to evaluate the delay/energy trade-off between different aspects relevant for the selection of suitable IoT communication technologies. If not noted otherwise, the energy values represent the chip with the lowest power consumption.

A. PROTOCOL DESIGN IMPACT

In this subsection, we will show how our models can be used to compare protocol design specific aspects of different technologies. As examples, we use the 6Lo header compression, acknowledgement schemes in SigFox and a comparison of different BLE versions.

1) 6LO HEADER COMPRESSION

The 6Lo specification includes IP header compression as an option to reduce the overhead due to header information [49]. This overhead can be quite significant, especially for small data amounts.

We added a feature into the 6Lo adaptation layer that allows to select, whether this compression is used or not. As an example, we run the models for BLE 5 and 802.15.4. Fig. 16 shows the comparison.

As expected, the header compression shows the most benefit for small packets with a few bytes or cases where the data is fragmented into few packets with a small final fragment. For higher application data amounts, the impact becomes smaller.

This result is expected, as it shows the defined behavior of the header compression option. However, even if it is specified, it does not mean that it is actually implemented in any case. For example in [50], Lenders report that RIOT does not support header compression currently, resulting in higher overhead and thus higher energy consumption. Our results clearly show, that this feature can help save energy and should be part of the software stack of constraint nodes.

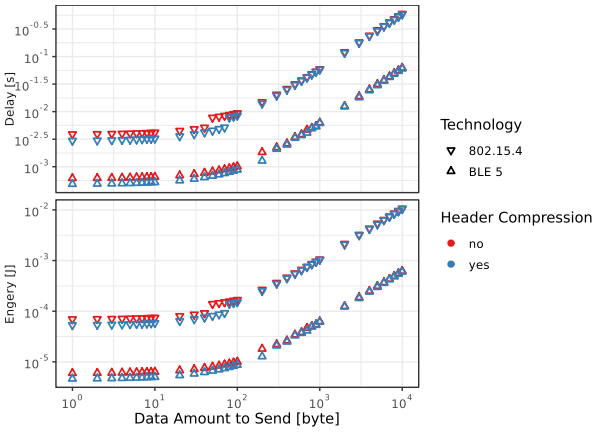


FIGURE 16. Impact of 6Lo header compression.

In contrast to other models that include the higher layers, we provide the option to choose between both configurations and thus enable the cost estimation for given implementations. This allows a more realistic overhead for 6LoWPAN based technologies.

2) SIGFOX ACKNOWLEDGEMENT SCHEMES

Due to its limited downlink capacity, SigFox [51] does not support acknowledgements for every uplink packet. If the transfer should still be confirmed, other schemes are needed. We modeled SigFox in a way that allows to use either no acknowledgement at all (this was also used for the evaluation in Fig. 14) or to use one final acknowledgement after the last uplink packet. Fig. 17 shows the comparison of both schemes, this time with a reduced range for the application data to transfer.

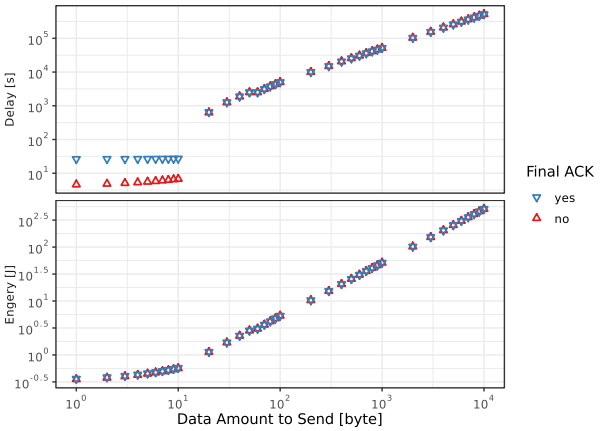


FIGURE 17. Impact of final ACK in SigFox.

Both schemes show a sudden increase in delay as soon as more than one packet has to be used due to the duty-cycle constraint, introducing long waiting times between subsequent packets. The jump is however not visible in the energy consumption, because we modeled the chip in a low-power sleep state during the waiting time. The usage of

an acknowledgement does however introduce a significant offset in the delay. In order to receive a downlink message, a SigFox transceiver has to wait for 20 s before receiving the acknowledgement. This corresponds to the standard and the measurements in [52]. Again the energy cost for the scheme shows no significant difference due to the usage of the low power mode for the waiting time.

Whether an acknowledgement is needed, has to be carefully assessed in SigFox networks. Besides the delay offset, the protocol operation does only support 8 downlink packets per day [52] limiting the availability further, even if only a final packet is to be confirmed. Acknowledgements are however a well-known protocol mechanism to minimize data loss/corruption during transmissions and could be used also to manage in-node memory (e. g. to free memory of successfully transfered data). If the application is fine with the additional offset, such a scheme might be good choice as the energy cost remains similar to a node without acknowledgements. To our knowledge this is the first model of SigFox covering the impact of a confirmed data transmission using acknowledgements.

3) COMPARISON OF BLE VERSIONS

The final protocol related comparison in this paper shows a comparison of three BLE versions 4.0, 4.2, and 5 based on their respective characteristics in Table 1. Versions 4.0 and 4.1 share the same characteristics and are described using one model. The same is true for versions 4.2 and 5 if the latter is operated on top of a 1 Mbit/s PHY layer. Delay values are therefore the same between these versions, while hardware specific energy consumption varies depending on the selected hardware.

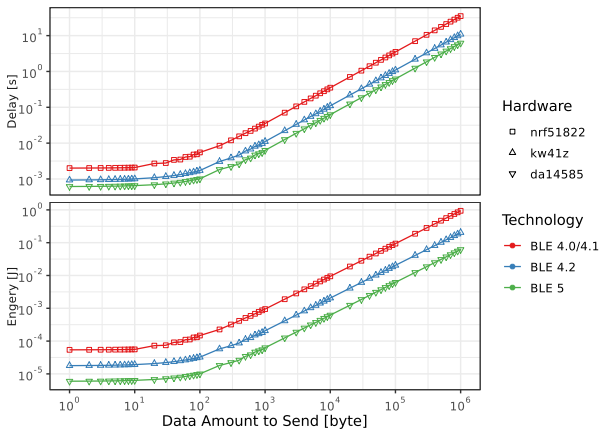


FIGURE 18. Comparison of different BLE versions.

Fig. 18 shows the comparison of the three BLE model variants. For each variant, we chose a chip supporting the base version of the models (e. g.the nrf51822 for BLE 4.0, the kw41z for BLE 4.2, and the da14585 for BLE 5). As expected, the version with the highest data rate also shows the lowest delay. That this version is also the one with the lowest energy consumption in this test is due to the fact, that the whole transfer simply is completed in less time thanks to the increased speed of the 2 Mbit/s PHY option in BLE 5. Besides that, in parallel to the evolution of the BLE versions, there has been a parallel development towards more energy efficient chips. Both aspects together lead to an increased efficiency for the transfers.

Our results show that there is a significant difference between the BLE versions in terms of delay and energy. The delay is a result of enhanced frame size and faster PHY of newer BLE versions. The reduced transmission durations also decrease the energy consumption since the transceiver is active for shorter durations. This effect is further increased by advances in the transceiver energy consumption for newer transceivers resulting in lower per state consumption.

Our models enable the comparison of all three versions, taking into account that currently available nodes can feature hardware supporting any of the discussed versions. In contrast, Morin *et al.* [11] focus on one version only but model multiple schemes for that. However, corresponding extensions of our framework to cover further operation modes are possible.

B. HARDWARE IMPACT ON ENERGY COST

Since the hardware is modeled as a separate class, it is possible to add multiple transceiver chips per technology and perform a comparison of hardware specific costs. In Fig. 19, we present an example comparison between five chips for 802.15.4 as listed in Table 3. The delay is not shown in this figure, because the protocol operation is considered the same in all cases.

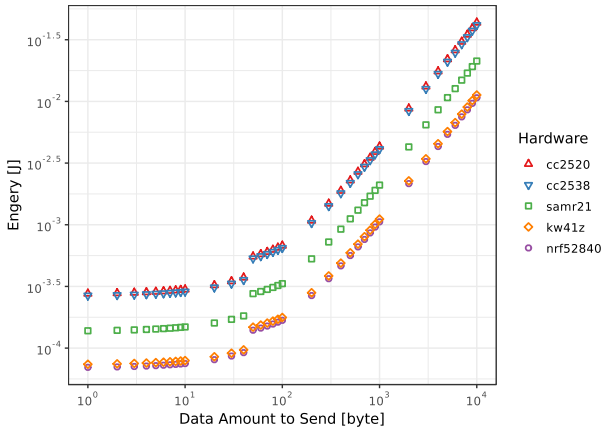


FIGURE 19. Impact of different 802.15.4 hardware on energy consumption.

By using different transceivers, we are able to visualize the trend towards more energy-efficient transceivers over the past years. The chip with the worst performance was developed 2007, while the best two were introduced 2016. The difference between the best and worst hardware is a factor four in this case.

In addition, our result shows that the chip selection is an important criterion for the selection of suitable IoT

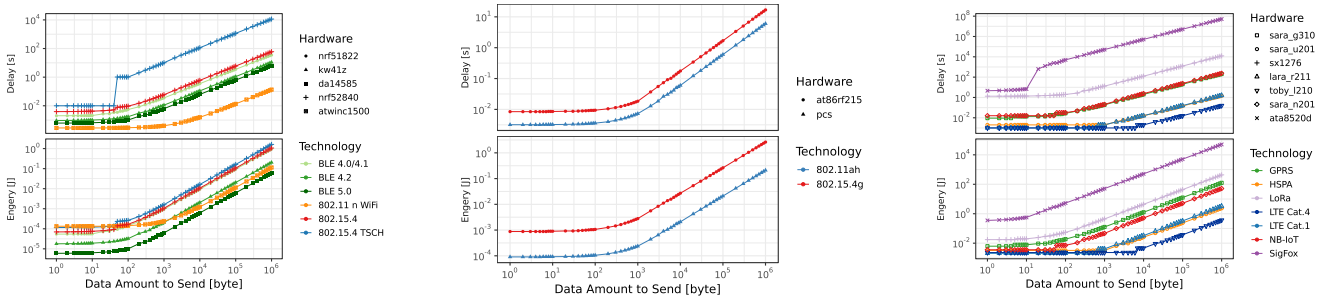


FIGURE 20. Comparison of modeled technologies (a) Short Range. (b) Medium Range. (c) Long Range.

communication technologies for energy constraint devices. A chosen technology might not perform as expected energy-wise, if the chosen hardware consumes too much energy.

Our tool supports this selection process by enabling an easy integration of further hardware for all technologies. Moreover, the results show that the values from a given hardware can differ quite significantly and therefore the usage of artificially optimized hardware as it is used in [11] should be discouraged. Instead, the realism of the model can be enhanced further by integrating measurement results for each transceiver state instead of the ideal datasheet values. Using this approach, the models can be used to predict the cost for a given real-world node.

C. COMPARISON OF TECHNOLOGIES

Next, we present a comparison between all supported technologies. This type of comparison enables an assessment of the suitability of different technologies for a given IoT scenario and the wide range of supported technologies sets our approach apart from previous studies. We use the range classification as mentioned in Section V-A mainly to enhance the visibility of the performance. Fig. 20 shows the delay and energy metrics for a single transfer of application data in the range from 1 Byte to 1 MByte.

When comparing the technologies, there are several points that are interesting especially if the goal is to select suitable technologies for a certain use case. These points show the trade-off decisions that have to be analyzed and assessed to choose a technology. At the same time, these points can lead to future research questions.

We start with a comparison of the short range technologies. As already discussed before, the delay introduced due to the default scheduling in TSCH is not scaling well with the data amount. The effect is not visible in the energy consumption, because the node is in sleep mode in all but its assigned slot. However, TSCH consumes slightly more energy than the traditional 802.15.4 version on the same hardware. This results from a higher activity within each time slot as compared to the 802.15.4 CSMA-based operation, at least under our ideal case. In non-ideal cases where multiple nodes compete for the medium leading to collisions this higher cost per slot will not have a significant impact. The schedule in TSCH with exclusively assigned slots then reduces the need for retransmissions and ensures a predictable delay. At the same time, the delay for CSMA-based 802.15.4 becomes unpredictable due to possible backoff times and retransmissions.

When comparing traditional 802.15.4 with the other technologies, it becomes obvious that its performance is very close to that of BLE 4.0. Both technologies were introduced nearly in parallel and have been developed further since they were first introduced. The BLE 5 however outperforms all 802.15.4 versions we considered here, due to the focus on enhanced data rates.

Besides these traditional short range WSN technologies, we added 802.11n into this category even if it is typically not mentioned in the context of small and low-power IoT devices. Our results show a reason for that clearly. 802.11n provides the lowest delay but the energy is comparatively high due to higher transceiver power consumption (cf. Table 3). This is a disadvantage especially for small data amounts that are common for traditional sensor nodes. As the application data amount increases to about 1 kByte, we observe an enhanced energy efficiency leading to the second best overall results. Therefore, WiFi has its right in this selection and might be an interesting candidate for devices that anyhow require more energy.

We modeled only two technologies for the medium range. In this case, 802.11ah clearly outperforms the 802.15.4g due to its higher PHY data rate (MCS 4) of 40 Mbit/s compared to 800 kbit/s. However, the evaluation for 802.11ah is done based on consumption values reported in [41] as datasheets for commercial chips were not available at the time writing this article. The energy consumption here has to be adjusted for real hardware.

Finally, we compare the long range technologies. As previously discussed, SigFox shows the worst performance of all technologies in terms of delay and energy consumption. While this was expected for higher data amounts, the high cost for small amounts of application data that can be transported within one SigFox transmission was surprising. SigFox has the highest energy consumption per transfer, even though the transceiver power consumption is in the range of older 802.15.4/BLE transceivers. The reason for this is the protocol operation of SigFox. In order to ensure some

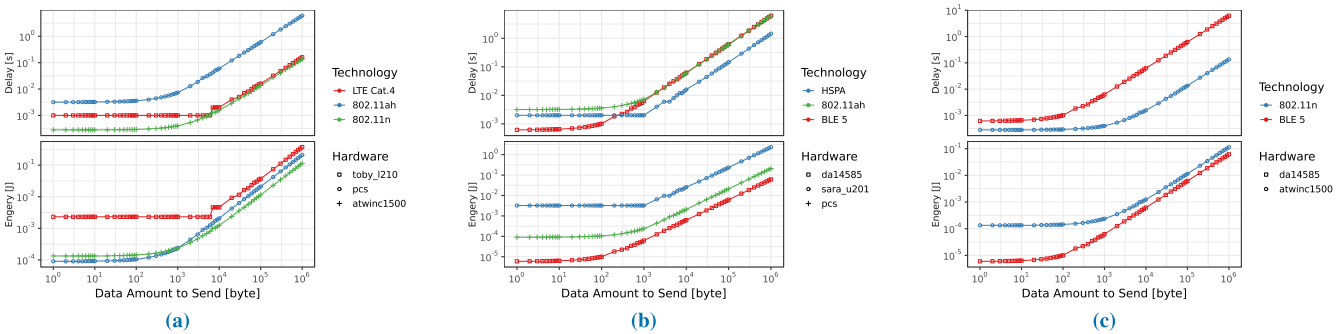


FIGURE 21. Comparison of different technology classes. (a) Lowest Delay. (b) Lowest Energy. (c) Lowest Delay vs. Lowest Energy.

reliability without acknowledgements, SigFox repeats each packet several times, resulting in the highest transmission delay for a single packet. With a delay for a single transmission of over 6 s the transceiver is active for a much longer time, resulting in the observed behavior.

Besides SigFox, the other two technologies that were specifically designed for IoT purposes have a relatively high delay and high energy consumption per transfer, too. This results mainly from low PHY data rates that increase the transmission delay and transceiver activity durations. In comparison to the short and medium range technologies, the three technologies have the advantage of the increased communication range. Compared to other long-range technologies, the improvements related to required overhead and power-saving mechanisms when nodes do not actively transmit should be beneficial but are not covered here.

While GPRS is the oldest technology in our comparison, the performance per transfer ends up in a medium range also explaining, why these modems are still popular for embedded solutions. Finally, the two LTE variants show the best performance here due to the increased PHY data rates that allow a fast completion of a data transfer. This is however only the case for larger data amounts. In case of small data amounts that correspond to the traditional sensor model, these technologies show a constant delay offset. The offset corresponds to one TTI interval that is the minimal transmission unit in LTE systems.

Next, we add another comparison considering the best technologies from each class against each other in order to see more trade-offs. Fig. 21 shows the corresponding diagrams for the technologies with the lowest delay (21a), the lowest energy consumption (21b), and the ones with the best combination of both metrics (21c) from each group.

When comparing the technologies with the lowest delay, all technologies stay below 10 ms for small data amounts. For higher data amounts, the technologies with high data rates show the expected advantage. LTE Cat.4 and 802.11n show a similar performance for higher data amounts but energy wise 802.11n is in advantage due to a much smaller power consumption of the transceiver (cf. Tables 3 and 4). Regarding the energy, 802.11ah is able to outperform both technologies for small data amounts.

When looking at the technologies with the best energy efficiency, the selection is different. Instead of 802.11n we now see BLE 5 from the short range technologies and HSPA from the long range technologies. Both have a reduced data rate compared to the respective technology with the shortest delay. HSPA shows a small improvement compared to LTE Cat.4 while requiring longer delay to complete the transfer due to a longer TTI.

As a result, we compare the BLE 5 and 802.11n in Fig. 21c as the two most efficient technologies. Here again, we observe the advantage of the higher data rate for 802.11n and the higher energy efficiency of BLE 5. However, depending on the use case, this plot shows that the increased delay of BLE 5 is compensated by a lower energy consumption for small data amounts while for larger data amounts 802.11n is preferred due to the lower delay at a similar energy consumption.

D. IMPACT OF TRAFFIC PATTERNS

One idea behind our models is to enable the evaluation of different traffic patterns other than the traditional periodic traffic. In this section, we introduce a case study showcasing this feature. If an application requires the sensor node to communicate data when certain criteria are met only, the actual transmissions are not periodic but rather event-based. We chose to model this, by using a Poisson arrival process. This allows us to generate randomly distributed events, that still follow a certain average arrival rate.

In our example, we use both a periodic sending strategy and two examples drawn from a Poisson process with the same average arrival rate. The arrival rate captures a scenario were in average 1 sample per hour is transfered. Fig. 22 shows the timely distribution of 100 generated events per traffic pattern.

The Poisson process results in a variation of the arrival times of subsequent events. This has two implications. At a given point in time, e.g. after two days, the number of events can be lower (green) or higher (blue) than for the periodic process. Besides that, the total duration to complete the 100 events differs as well.

We sampled the Poisson values ensuring that at least one transmission of a given data amount can be completed within the minimum inter arrival time of the samples at hand.

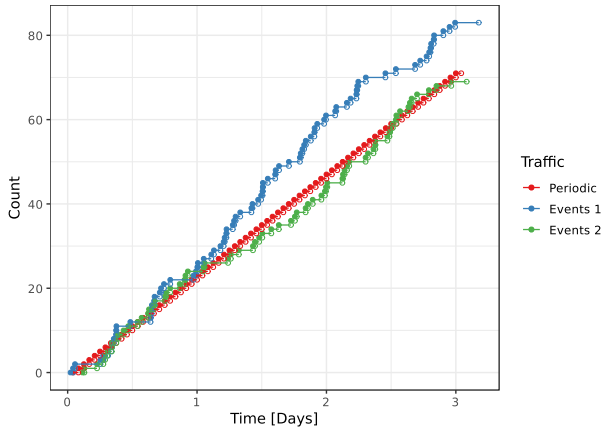


FIGURE 22. Time distribution of sampled events for each traffic pattern.

Based on that, we then calculated the accumulated energy consumption for the most energy efficient BLE 4.0 transceiver for three different constant data amounts per transmission for each traffic pattern. Similar to the lifetime estimation in Fig. 15, we use the sleep mode of the transceiver to calculate the energy consumption between the individual transmissions. Fig. 23 shows the corresponding results.

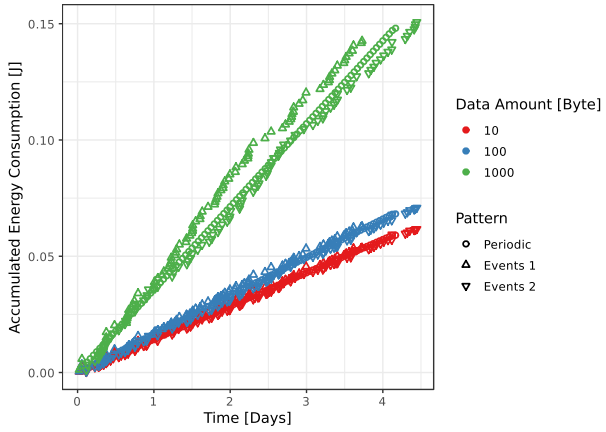


FIGURE 23. Accumulated energy consumption with different traffic patterns.

The energy consumption follows the event distribution in all cases, as expected. As a result, we observe the same variations as for the event distribution itself. On one hand, a given number of events can consume more or less energy depending on the timely context and on the other hand, a different number of events can be handled with a certain energy budget.

By using or models for single data transfers and additional calculations for the event generation were are able to analyze both questions for a given technology. Besides that, our approach enables the evaluation of scenarios where the data amount in such an event-based scheme is not constant, e.g. is the sensing is periodic but the communication is not and data gets accumulated as well. In such cases, the call to the functions is executed with the respective data amount when the event takes place. These use cases are more realistic regarding the traffic and transceiver activity compared to traditional analytical models.

E. IMPACT OF MULTIPLE NODES

Finally, we would like to address more complex scenarios as described in Section II. There, we stated that in a distributed IoT system, nodes at the sensing level can have several roles/tasks depending on the actual deployment. Possible roles include the traditional sensor that produces data and sends this towards a sink but also hybrid forms, where the node acts as relay for other nodes in its surrounding. In both cases, the simplified two-node model we propose in this work covers only a limited extend of possible interactions, as traffic from further neighboring nodes is not considered.

Depending on the medium access scheme at hand, traffic from neighboring nodes can however increase the latency if the channel is busy or resources get distributed between multiple nodes. In addition, collisions between packets can occur triggering retransmissions that consume additional time and energy. While these aspects are not covered in our models directly, we are still able to perform some analyses regarding a common collision domain for neighboring nodes.

Especially, if nodes do not send their data at the same time, we can estimate how many nodes can use a common collision domain (e.g. in a star topology talking to a common relay) without overlapping each others transmissions for a given data amount. This gives insights on the minimum required sending interval as well as the maximum data amount at application layer to achieve this. Similarly, we can analyze the traffic patterns of relay nodes that are part of the last mile network, e.g. in all ad-hoc style networks. This requires the energy to receive and forward data from all child nodes and that requires to optionally send the data from the relay as well, if it follows a hybrid approach. Since we model an individual data transfer, retransmissions are only possible if they are triggered from the application. Any automatic lower layer retransmissions to recover from transmission errors is however not covered, leading to an overestimation of the required time and energy consumption for retransmissions. However, adding a certain error percentage to the transmissions as done in [11] is possible.

Using these approaches, we are also able to cover other use cases than just the simple two node case and provide more flexibility to the analysis of energy cost at node-level depending on the actual role of the node in the network. This sets our work apart from other analytical models while not achieving the flexibility of simulations. As a comparison framework, that provides more technologies than previously covered, it is however a valuable intermediate tool between both worlds.

VII. CONCLUSIONS

In this paper, we presented our approach to estimate delay and energy consumption of a single data transfer covering

a wide range of IoT last mile communication technologies. All models together provide a valuable tool for both the system design and the research and development of new communication standards. The modular design allows the integration of further technologies or other configurations.

Based on the presented example comparisons, we showed that our models provide a useful tool that allows to estimate both delay and energy consumption for the included technologies. Using this tool, trade-offs between delay and energy can be analyzed. This is not limited to the comparison between technologies. Comparisons between different hardware and operation options are also possible.

The per transfer modeling allows more flexibility than traditional models that focus on periodic traffic only. Using additional wrapper functions on top of the current implementation allows estimations for various traffic patterns that become more relevant for preprocessing or event-based sensing applications. As such, it provides valuable insights on energy-efficient communication required for various application fields.

As future work, we plan to apply the models to several specific IoT scenarios and use cases in order to identify a set of suitable technologies. In addition, we plan to evaluate the impact of protocol introduced overhead and retransmission schemes on the communication cost. Finally, further extensions of the models are planned. These include among others the introduction of non-ideal transmission scenarios and packet error probabilities as well as extensions considering collision domains for those technologies applicable. As alternatives to the current stack setup, we plan to add further protocols as for example TCP.

APPENDIX A
LIST OF ABBREVIATIONS

BLE	Bluetooth Low Energy
CSMA	Carrier Sense Multiple Access
CSMA-CA	Carrier Sense Multiple Access - Collision Avoidance
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HSPA	High-Speed Packet Access
IoT	Internet of Things
IP	Internet Protocol
ISM	Industrial Scientific and Medical
LAN	Local Area Network
LoRa	Long Range
LP-WAN	Low Power Wide Area Networks
LTE	Long Term Evolution
MAC	Medium Access Control
MAN	Metropolitan Area Network
MCS	Modulation and Coding Scheme
MIMO	Multiple Input Multiple Output
MTU	Maximum Transmission Unit
M2M	Machine-to-Machine
NB-IoT	Narrow Band IoT
NFC	Near Field Communication
OFDM	Orthogonal Frequency Division Multiplex
OS	Operating System
OSI	Open Systems Interconnection
PAN	Personal Area Network
PDCP	Packet Data Convergence Protocol
PHY	Physical Layer
RFID	Radio Frequency Identification
RLC	Radio Link Control
SF	Spreading Factor
TB	Transport Block
TCP	Transmission Control Protocol
TDD	Time Division Duplex
TSCH	Time Synchronized Channel Hopping
TTI	Transmission Time Interval
UDP	User Datagram Protocol
WAN	Wide Area Network
WSN	Wireless Sensor Network
3GPP	3rd Generation Partnership Project
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks

APPENDIX B
LIST OF NOTATIONS AND VARIABLES

d	application data amount to transfer
$data_{pdcp}$	total amount of data after PDCP processing, including PDCP and upper layer overhead.
$E_{<X>}$	total energy consumption of component $<X>$, where $<X>$ represents the node n itself or the sensing s, processing p, and communication c components.
$E_c(d, hw)$	total energy consumption of communication to transfer data amount d using transceiver hardware hw.
$f_{pl,te}$	payload size of an 6LoWPAN fragment for a given underlaying technology te.
hw	a selected transceiver hardware used for energy calculations.
ip_{he}	header size of an IPv6 packet.
$ip_{he,comp}$	header size of an IPv6 packet with header compression.
ip_{mtu}	maximum packet size of an IPv6 packet, including the header.
ip_{pl}	payload size of a full-sized IPv6 packet.
$ip_{pl,f}$	payload size of the final IPv6 packet.
mac_{pl}	payload size of a maximum-sized MAC frame.
$n_{ip}(d)$	total number of IPv6 packets required to transfer data amount d.
$n_{rlc}(d)$	number of RLC transmissions required to transfer data amount d.
$n_T(d, te)$	number of individual transmissions required to transfer data amount d using technology te.
$n_{6lo_m,ip}$	number of maximum-sized 6LoWPAN fragments per full-sized IPv6 packet.

n_{6lo_m,ip_l}	number of maximum-sized 6LoWPAN fragments of the final, potentially shorter, IPv6 packet.
$n_{6lo_m,te}(d)$	total number of maximum-sized 6LoWPAN fragments required to transfer data amount d using technology te.
$n_{6lo,te}(d)$	total number of 6LoWPAN fragments required to transfer data amount d using technology te.
$P_{<X>}(hw)$	power consumption of transceiver hardware hw in state $<X>$. Supported states: transmit tx, receive, rx idle i, or sleep s.
$pdcp_{he}$	header size of a PDCP packet.
$pdcp_{pl}$	payload size of a PDCP packet.
rlc_{he}	header size of a RLC packet.
$t_{<X>}(d)$	time spend in state $<X>$ to transfer data amount d. $<X>$ denotes the total communication delay t_c or the time for each transceiver state: transmit tx, receive, rx idle i, or sleep s.
udp_{he}	header size of an UDP packet.
$6lo_{he,te}$	header size of a 6LoWPAN fragment for a given underlaying technology te.
$6lo_{pl,<X>,te}$	payload size of a 6LoWPAN fragment for a given underlaying technology te. $<X>$ denotes a maximum-sized m fragment, the size of the last l fragment required for a full-sized IPv6 packet, and the final f fragment of the final, potentially shorter, IPv6 packet.

REFERENCES

[1] F. Vannieuwenborg, S. Verbrugge, and D. Colle, "Choosing IoT-connectivity? A guiding methodology based on functional characteristics and economic considerations," *Trans. Emerg. Telecommun. Technol.*, vol. 29, no. 5, p. e3308, 2018.

[2] M. Imran, K. Shahzad, N. Ahmad, M. O'Nils, N. Lawal, and B. Oelmann, "Energy-efficient SRAM FPGA-based wireless vision sensor node: SENTIOF-CAM," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 12, pp. 2132–2143, Dec. 2014.

[3] K. Shahzad and M. O'Nils, "Condition monitoring in industry 4.0-design challenges and possibilities: A case study," in *Proc. Workshop Metrol. Ind. 4.0 IoT*, Apr. 2018, pp. 101–106.

[4] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.

[5] R. Sanchez-Iborra and M.-D. Cano, "State of the art in LP-WAN solutions for industrial IoT services," *Sensors*, vol. 16, no. 5, p. 708, 2016.

[6] F. Samie, L. Bauer, and J. Henkel, "IoT technologies for embedded computing: A survey," in *Proc. Int. Conf. Hardw./Softw. Codesign Syst. Synth. (CODES+ISSS)*, Oct. 2016, pp. 1–10.

[7] M. R. Palattella *et al.*, "Internet of Things in the 5G era: Enablers, architecture, and business models," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, pp. 510–527, Mar. 2016.

[8] M. Carratù, M. Ferro, V. Paciello, A. Pietrosanto, and P. Sommella, "Performance analysis of wM-bus networks for smart metering," *IEEE Sensors J.*, vol. 17, no. 23, pp. 7849–7856, Dec. 2017.

[9] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proc. 1st Int. Conf. Simulation Tools Techn. Commun., Netw. Syst. Workshops (SIMUTools)*, Mar. 2008, Art. no. 60.

[10] *Network Simulator 3 (NS-3)*. Accessed: Oct. 17, 2018. [Online]. Available: http://www.nsnam.org/

[11] É. Morin, M. Maman, R. Guizzetti, and A. Duda, "Comparison of the device lifetime in wireless networks for the Internet of Things," *IEEE Access*, vol. 5, pp. 7097–7114, 2017.

[12] S. Krug, A. Schreiber, and M. Rink, "Poster: Beyond static sending intervals for sensor node energy estimation," in *Proc. 12th Workshop Challenged Netw. (CHANTS)*, 2017, pp. 39–41.

[13] I. Shallari, S. Krug, and M. O'Nils, "Architectural evaluation of node: Server partitioning for people counting," in *Proc. 12th Int. Conf. Distrib. Smart Cameras (ICDSC)*, 2018, p. 1.

[14] *Near Field Communication Interface and Protocol (NFCIP-1)*, Ecma International, Geneva, Switzerland, 2013.

[15] S. Dominikus, M. Aigner, and S. Kraxberger, "Passive RFID technology for the Internet of Things," in *Proc. Int. Conf. Internet Technol. Secured Trans. (ICITST)*, Nov. 2010, pp. 1–8.

[16] X. Jia, Q. Feng, T. Fan, and Q. Lei, "RFID technology and its applications in Internet of Things (IoT)," in *Proc. 2nd Int. Conf. Consum. Electron., Commun. Netw. (CECNet)*, Apr. 2012, pp. 1282–1285.

[17] D. de Donno, L. Catarinucci, and L. Tarricone, "RAMSES: RFID augmented module for smart environmental sensing," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 7, pp. 1701–1708, Jul. 2014.

[18] C. Gomez, J. Paradells, C. Bormann, and J. Crowcroft, "From 6LoWPAN to 6Lo: Expanding the universe of IPv6-supported technologies for the Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 148–155, Dec. 2017.

[19] *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE Standard 802.15.4-2011, 2011.

[20] H. Toepfer, E. Chervakova, M. Goetze, T. Hutschenreuther, B. Nikolić, and B. Dimitrijević, "Application of wireless sensors within a traffic monitoring system," in *Proc. 23rd Telecommun. Forum Telfor (TELFOR)*, Belgrade, Serbia, Nov. 2015, pp. 236–241.

[21] D. De Guglielmo, S. Brienza, and G. Anastasi, "IEEE 802.15.4e: A survey," *Comput. Commun.*, vol. 88, pp. 1–24, Aug. 2016.

[22] *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, 2012.

[23] *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 3: Physical Layer (PHY) Specifications for Low-Data-Rate, Wireless, Smart Metering Utility Networks*, IEEE Standard 802.15.4g-2012, 2012.

[24] *Bluetooth Specification Version 4.0*, Bluetooth SIG, Kirkland, WA, USA, 2010.

[25] *Bluetooth Core Specification Version 5.0*, Bluetooth SIG, Kirkland, WA, USA, 2016.

[26] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez, *IPv6 Over Bluetooth(R) Low Energy*, document RFC 7668, Internet Engineering Task Force, Oct. 2015. [Online]. Available: http://www.ietf.org/rfc/rfc7668.txt

[27] J. Rüth, F. Schmidt, M. Serror, K. Wehrle, and T. Zimmermann, "Communication and networking for the industrial Internet of Things," in *Industrial Internet of Things*. Cham, Switzerland: Springer, 2017, pp. 317–346.

[28] S. Petersen and S. Carlsen, "WirelessHART versus ISA100.11a: The format war hits the factory floor," *IEEE Ind. Electron. Mag.*, vol. 5, no. 4, pp. 23–34, Dec. 2011.

[29] *IEEE Standard for Information Technology–Local and Metropolitan Area Networks–Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput*, IEEE Standard 802.11n-2009, 2009.

[30] *IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems–Local and Metropolitan Area Networks–Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 2: Sub 1 GHz License Exempt Operation*, IEEE Standard 802.11ah-2016, 2016.

[31] A. Hazmi, J. Rinne, and M. Valkama, "Feasibility study of IEEE 802.11ah radio technology for IoT and M2M use cases," in *Proc. Globecom Workshops (GC Wkshps)*, Dec. 2012, pp. 1687–1692.

[32] B. Vejlgaard, M. Lauridsen, H. Nguyen, I. Z. Kovács, P. Mogensen, and M. Sørensen, "Coverage and capacity analysis of Sigfox, LoRa, GPRS, and NB-IoT," in *Proc. 85th Veh. Technol. Conf. (VTC)*, Jun. 2017, pp. 1–5.

[33] H. Holma, A. Toskala, K. Ranta-Aho, and J. Pirskanen, "High-speed packet access evolution in 3GPP release 7," *IEEE Commun. Mag.*, vol. 45, no. 12, pp. 29–35, Dec. 2007.

[34] R. Ratasuk, B. Vejlgaard, N. Mangalvedhe, and A. Ghosh, "NB-IoT system for M2M communication," in *Proc. Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2016, pp. 428–432.
[35] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: The rising stars in the IoT and smart city scenarios," *IEEE Wireless Commun.*, vol. 23, no. 5, pp. 60–67, Oct. 2016.
[36] R. Kirichek, V.-D. Pham, A. Kolechkin, M. Al-Bahri, and A. Paramonov, "Transfer of multimedia data via LoRa," in *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*. Cham, Switzerland: Springer, 2017, pp. 708–720.
[37] B. Martinez, M. Montón, I. Vilajosana, and J. D. Prades, "The power of models: Modeling power consumption for IoT devices," *IEEE Sensors J.*, vol. 15, no. 10, pp. 5777–5789, Oct. 2015.
[38] S. Krug. (2018). *MATLAB Functions for Cost Estimation of IoT Data Transfers*. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:miun:diva-34861
[39] H. Ayers, P. Crews, H. Teo, C. McAvity, A. Levy, and P. Levis. (2018). "Design considerations for low power Internet protocols." [Online]. Available: https://arxiv.org/abs/1806.10751
[40] A. Larmo, M. Lindström, M. Meyer, G. Pelletier, J. Torsner, and H. Wiemann, "The LTE link-layer design," *IEEE Commun. Mag.*, vol. 47, no. 4, pp. 52–59, Apr. 2009.
[41] A. Ba *et al.*, "A 4mW-RX 7mW-TX IEEE 802.11ah fully-integrated RF transceiver," in *Proc. Radio Freq. Integr. Circuits Symp. (RFIC)*, Jun. 2017, pp. 232–235.
[42] X. Vilajosana *et al.*, "A realistic energy consumption model for TSCH networks," *IEEE Sensors J.*, vol. 14, no. 2, pp. 482–489, Feb. 2014.
[43] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister, "OpenMote: Open-source prototyping platform for the industrial IoT," in *Proc. Int. Conf. Ad Hoc Netw.* Cham, Switzerland: Springer, 2015, pp. 211–222.
[44] E. Baccelli *et al.*, "RIOT: An open source operating system for low-end embedded devices in the IoT," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4428–4440, Dec. 2018.
[45] M. Lenders *et al.* (2018). "Connecting the world of embedded mobiles: The RIOT approach to ubiquitous networking for the Internet of Things." [Online]. Available: https://arxiv.org/abs/1801.02833
[46] L. F. D. Carpio, P. Di Marco, P. Skillermark, R. Chirikov, and K. Lagergren, "Comparison of 802.11ah, BLE and 802.15.4 for a home automation use case," *Int. J. Wireless Inf. Netw.*, vol. 24, no. 3, pp. 243–253, 2017.
[47] M. Spörk, C. A. Boano, M. Zimmerling, and K. Römer, "BLEach: Exploiting the full potential of IPv6 over BLE in constrained embedded IoT devices," in *Proc. SenSys*, 2017, Art. no. 2.
[48] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the limits of LoRaWAN," *IEEE Commun. Mag.*, vol. 55, no. 9, pp. 34–40, Sep. 2017.
[49] J. Hui and P. Thubert, *Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks*, document RFC 6282, Internet Engineering Task Force, Sep. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6282.txt
[50] M. Lenders, "Analysis and comparison of embedded network stacks," M.S. thesis, Dept. Comput. Sci., Freie Univ. Berlin, Berlin, Germany, 2016.
[51] *SigFox—Technical Overview*, SigFox, Toulouse, France, May 2017.
[52] D. M. Hernandez, G. Peralta, L. Manero, R. Gomez, J. Bilbao, and C. Zubia, "Energy and coverage study of LPWAN schemes for industry 4.0." in *Proc. IEEE Int. Workshop Electron., Control, Meas., Signals Appl. Mechatronics (ECMSM)*, Donostia-San Sebastian, Spain, May 2017, pp. 1–6. doi: 10.1109/ECMSM.2017.7945893.

SILVIA KRUG received the B.Sc. degree in computer science from the University of Applied Sciences Darmstadt, Germany, and the M.Sc. degree in computer engineering and the Dr.Ing. degree in communication networks from Technische Universtät Ilmenau, Germany, in 2013 and 2017, respectively. She currently holds a postdoctoral position with the Embedded Systems Design Research Group, Mid Sweden University, Sweden. Her research interests include communication aspects for wireless sensor systems with a focus on energy-efficiency and design considerations for the distributed IoT systems.

MATTIAS O'NILS received the B.S. degree in electrical engineering from Mid Sweden University, Sundsvall, Sweden, in 1993, and the Licentiate and Ph.D. degrees in electronics system design from the Royal Institute of Technology, Stockholm, Sweden, in 1996 and 1999, respectively. He is currently a Professor with the Department of Electronics Design and leads a research group in embedded systems design at Mid Sweden University. His current research interests include design methods and implementation of embedded systems, especially in the implementation of real-time video processing systems.