


IoT Bench: Towards a Benchmark for Low-power Wireless Networking

Conference Paper**Author(s):**

Boano, Carlo A.; Duquennoy, Simon; Förster, Anna; Gnawali, Omprakash; [Jacob, Romain](#) ; Kim, Hyung-Sin; Landsiedel, Olaf; Marfievici, Ramona; Mottola, Luca; Picco, Gian Pietro; Vilajosana, Xavier; Watteyne, Thomas; Zimmerling, Marco

Publication date:

2018-04

Permanent link:

<https://doi.org/10.3929/ethz-b-000256517>

Rights / license:

[In Copyright - Non-Commercial Use Permitted](#)

Originally published in:

<https://doi.org/10.1109/CPSBench.2018.00013>

IoT Bench: Towards a Benchmark for Low-power Wireless Networking

Carlo Alberto Boano, Simon Duquennoy, Anna Förster, Omprakash Gnawali, Romain Jacob, Hyung-Sin Kim, Olaf Landsiedel, Ramona Marfievici, Luca Mottola, Gian Pietro Picco, Xavier Vilajosana, Thomas Watteyne, and Marco Zimmerling

www.iotbench.ethz.ch

Abstract—Unlike other fields of computing and communications, low-power wireless networking is plagued by one major issue: the absence of a well-defined, agreed-upon yardstick to compare the performance of systems, namely, a benchmark. We argue that this situation may eventually represent a hampering factor for a technology expected to be key in the Internet of Things (IoT) and Cyber-physical Systems (CPS). This paper describes a recent initiative to remedy this situation, seeking to enlarge the participation from the community.

I. INTRODUCTION

Low-power wireless communication is a key enabling technology for the Internet of Things (IoT) and Cyber-physical Systems (CPS). Network stacks supporting multi-hop communications have received particular attention, as they bring flexibility in the physical deployment of devices. The last decade has seen tremendous improvement in the performance of these network stacks: near-perfect reliability is now a reality, and can be achieved with extremely low latency and energy consumption.

Problem: Lack of confidence in results. Despite the performance claims in several academic papers and commercial system, even in the many cases where these claims are substantiated by rigorous and realistic evaluations, the extent to which the results for one system hold in the setup of another is still unclear at best.

A simple example about periodic data collection, arguably the most-studied traffic pattern, illustrates the problem. Fig. 1 shows manually-gathered statistics from 32 papers published in flagship conferences in recent years (ACM SenSys, IEEE/ACM IPSN, INFOCOM, MASS, DCOSS and EWSN 2010 to 2015). The chart plots two simple parameters known to have significant impact on system performance: the number of nodes in the tested network and the inter-packet interval (IPI).

C.A. Boano is with Graz University of Technology, Austria.
S. Duquennoy is with RISE SICS, Sweden.
A. Förster is with the University of Bremen, Germany.
O. Gnawali is with University of Houston, TX, USA.
R. Jacob is with ETH Zurich, Switzerland.
H.-S. Kim is with University of California at Berkeley, CA, USA.
O. Landsiedel is with Chalmers University of Technology, Sweden.
R. Marfievici is with Cork Institute of Technology, Ireland.
L. Mottola is with Politecnico di Milano, Italy and RISE SICS, Sweden.
G.P. Picco is with University of Trento, Italy.
X. Vilajosana is with Universitat Oberta de Catalunya, Catalonia, Spain.
T. Watteyne is with Inria, France.
M. Zimmerling is with TU Dresden, Germany.

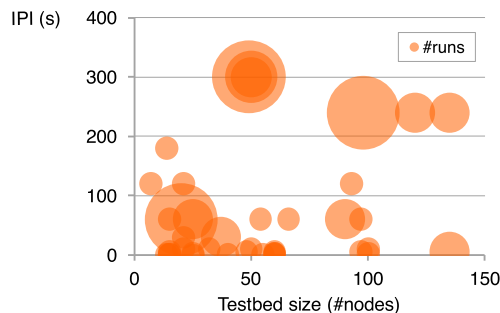


Fig. 1. Comparing the experimental setup of systems supporting periodic data collection (2010–2015).

It is easy to see that different setups cover different areas of the chart. Do the results reported for a given combination hold for another? In practice, the divergence across experimental setups is even higher, as the application and environmental parameters affecting performance are many, and in many cases include the configuration of the protocols under consideration.

This situation is not just methodologically flawed from a scientific standpoint; it has also distorting effects on industrial practice and its connection with research. For instance, one of the reasons companies are reportedly reluctant to embrace multi-hop low-power wireless due to the *perception* that it is too unreliable, or otherwise not performant. Many research papers and even commercial systems claim otherwise; however, due to the aforementioned fragmentation of experimental setups, it is hard to overcome this perception.

Solution. To overcome this status quo, we need a common, agreed-upon yardstick against which the performance of systems can be measured—in other words, a *benchmark*.

Benchmarks have been designed and routinely used in other communities, including database systems [7], robotics [28], and image processing [6], to compare research and commercial products. A benchmark typically defines input parameters representative of several application cases, output metrics quantifying the performance of the benchmarked systems, and possibly the required experimental setup. For instance, a set of representative datasets (e.g., data or image) may be provided as input, and the speed or accuracy in performing some function (e.g., a specific database query or image recognition) may be measured as output.

Therefore, there is a lot of previous expertise one can draw upon to define a benchmark for low-power wireless

networking. Nevertheless, what distinguishes this field from, say, image processing, is the fact that system performance is affected by real-world dynamics, such as the vagaries of low-power wireless links. The number and density of nodes defining the network topology matters, but also the characteristics of the environment; factors like indoor vs. outdoor scenarios, the presence of interference, or variation in physical parameters like temperature and humidity, may greatly affect connectivity and system performance. Fields like robotics and control face similar challenges. As a consequence, the *experimental environment plays a central role in the definition of a benchmark for low-power wireless networking.*

Goals. The overarching goal of our benchmarking effort is to bring a standard and consistent way to test and compare low-power wireless networking protocols. We consider main stakeholders for our benchmark design: *i)* researchers who want to evaluate new protocols against older ones or baselines, *ii)* companies to showcase the advantage of their protocols, and *iii)* customers to help decide objectively strengths of protocols. A set of common goals and trade-offs exist that reasonably satisfies the needs of three stakeholders:

- *The benchmark should be reproducible across experimental settings.* To achieve this goal, a benchmark should contain sufficient context information to recreate the experiments.
- *The benchmark should be simple and minimalist.* This goal ensures that a benchmark is applicable as broadly as possible.
- *The benchmark should suggest sufficient and appropriate experiments and metrics to improve the state of experimentation.* Thus, benchmarking is also an opportunity to educate the community on best practices.
- *The benchmark should allow for consistent comparisons of low power wireless protocols across key metrics.*

Contribution and road-map. In this paper, we outline the challenges that must be addressed to realize the vision above. We begin by discussing the approach (§II) we follow in designing this benchmark. This starts from an implementation-agnostic definition of application profiles and performance metrics (§III) and aims to ultimately achieve the implementation (§IV) of an experimental setup enabling the automated and reproducible execution of the benchmark. Finally, we discuss the current status of our initial efforts (§V) along with some key elements we expect to determine the practical adoption of the benchmark; these are the elements based on which we conclude the paper with an outlook (§VI) towards future activities.

II. DESIGN SPACE AND APPROACH

In designing our benchmark, we face conflicting requirements. On one hand, as illustrated in Fig. 2, we would like the benchmark to be as general and representative as possible, i.e., encompassing a wide array of application use cases, technical approaches, and experimental setups. On the other hand, the desire to enable accurate, fair, and repeatable comparison pushes towards a very detailed and prescriptive experimental setup, possibly yielding to the (undesirable) extreme of forcing

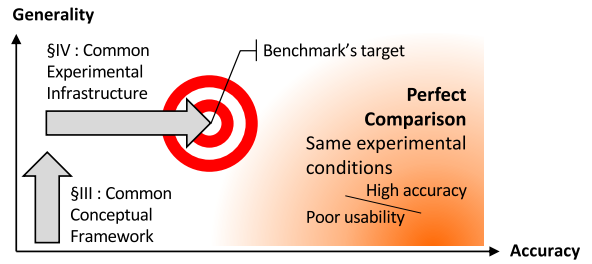


Fig. 2. The design space of protocol comparison. *Our approach aims for increasing comparison accuracy while preserving generality and usability of the benchmark.*

the comparison to take place in exactly the same experimental conditions, e.g., the very same testbed. Question is how to reconcile these conflicting goals.

The first step is the definition of parameters, metrics, and profiles. As further detailed in §III, together these effectively provide a *specification* of the input provided to an experiment, the outputs measured, and the conditions observed during its execution, analogously to the benchmark definitions in other fields. At this level of abstraction, metrics and profiles provide the *common conceptual framework* that constitutes the minimal foundation for a benchmark. Indeed, without a common definition of these aspects, the whole notion of comparability is at stake. Consequently, the agreed-upon definition of metrics and profiles can also be immediately useful as a way to *i)* guide system evaluation by clearly identifying what portion of the configuration space is being covered and, as a consequence *ii)* qualify the results obtained, by separating the conditions in which they hold from those for which nothing can be said.

In their more abstract incarnation, parameters, metrics and profiles are technology-agnostic; the specific radio technology can be regarded as one of the attributes of the profile describing an experiment. Further, they are also implementation-agnostic, which means that they are applicable to different evaluation methods, e.g., simulation vs. testbed experiments. The level of detail included in the definition of profiles defines the trade-offs between generality and reproducibility, e.g., by progressively including in the profiles the information necessary to reproduce the environmental conditions.

Specifying experimental parameters, however, may be too weak without a way to enforce them, and at the same time pose too much burden on the researcher performing the experiments. For this reason, our aim is also to define, design, and implement a *common experimental infrastructure* supporting the execution of experiments based on our benchmark suite. In this vision, an entire toolchain is available to researchers and practitioners, which simplifies and automates the selection, deployment, and execution of benchmarks, along with their sharing and dissemination to the community. Several challenges are associated with this goal. The technical ones are discussed in detail in §IV, both for simulation and testbed experiments. However, a more subtle challenge is to prevent a readily available infrastructure from effectively “locking” a research community to a common hardware/software platform, effectively stifling innovation, as we further discuss in §VI. In this respect, however, the approach illustrated in Fig. 2 also helps, as in principle it enables the definition of different

degrees of compliance and reproducibility to a benchmark suite, namely, ranging from the use of the sole specification to the use of the same experimental infrastructure.

III. IOTBENCH PARAMETERS, METRICS AND PROFILES

We seek to provide a well-defined structure to IoTBench, a benchmark for IoT, while not limiting the scope of its applicability.

Parameters and metrics. We define key classes of parameters and metrics that we consider in the definition of IoTBench. A metric is a measurable quantity representative of some specific facet of a benchmark, but not necessarily originating from a deterministic process.

- 1) *Input* parameters: are quantities that users feed as input to a certain run of a benchmark. They are completely under the user's control and should, in principle, be independent among themselves and of the protocol involved in a run of a benchmark. Examples are the number of nodes involved in a run, or the traffic load.
- 2) *Output* metrics: are quantities that users obtain as the result of running a benchmark. They are completely determined by the behavior of a given protocol given a fixed set of input metrics, and thus out of the user's control. Examples are the packet delivery rate or energy consumption of a protocol.
- 3) *Observed* metrics: are quantities that users also obtain as the result of a run of a benchmark, typically to characterize the environment when a benchmark was run. They are indeed necessary to enable fair comparison between different protocols. Examples are the level of external interference or people's presence during a run.

It is therefore useful to map metrics commonly found in experimental studies of protocols to the three classes above. These, in turn, may be grouped as:

- *System* parameters: such as network topology, link qualities, and mobility patterns. Depending on the nature of the metric, these may be either *input* to a run of a benchmark, or *observed*. For example, network topologies are usually input to a run of a benchmark, at least to the extent possible in a given environment. Link qualities, however, do not enjoy direct control and thus may only be *observed* during a run, to check whether different runs are comparable in terms of wireless dynamics.
- *Traffic* parameters: such as traffic load, traffic pattern, and number of sources. These metrics should always be *input* to a run. Most importantly, a necessary condition for a fair comparison between different protocols is that their behavior is determined by the exact same set of input parameters. Differences in these sets may already question the soundness of the comparison.
- *External* metrics, such as external interference and people's presence are eminently *observed*. Similar to the observation above for traffic parameters, a necessary condition for a fair comparison between different protocols is that these observed metrics are comparable across runs of different protocols.

- *Performance* metrics, such as packet delivery or latency, and energy consumption are eminently *output* as they typically represent the results of a run of a benchmark that users are interested in. They represent the quantitative basis to compare different protocols, and must solely be determined by a protocol's behavior given certain assignments to all input parameters.
- *Experiment* parameters such as the number of repetitions and the acceptable standard deviation of key performance metrics are also typically *input*. Rather than directly impacting the output metrics, they most often determine the scope of applicability of the output metrics; for example, by identifying the conditions to exclude certain results from the analysis.

The metric classification we describe is instrumental in creating combinations of metrics that uniquely characterize a certain run of a benchmark.

Profiles. A profile is a possibly partial assignment of concrete values to input parameters, and a precise definition of observed and output metrics to be measured. In other words, a profile is a set of points in the n -dimensional space determined by n input parameters.

A profile should ideally map to a concrete application or domain. For example, we may define a *low-rate data collection* profile representing applications such as environmental monitoring, with the following structure:

- Input parameters: #nodes: 100, #sources: 99, #destinations: 1, traffic load: from .1 msg/min to 1 msg/min.
- Observed metrics: link qualities and external interference.
- Output metrics: packet delivery rate, end-to-end packet latency, and average energy consumption.

It goes without saying that the definition of each and every metric is a challenge per se. For example, even seemingly simple or commonly employed metrics may be defined differently. Energy consumption, for instance, may refer to the average energy consumption of all nodes in the network, or only to a subset of nodes deemed significant. Once the structure of metrics and profiles is in place, a benchmark definition should then provide specific semantics of all metrics involved, similar to existing literature in other fields [6], [7], [28].

IV. IMPLEMENTATION

IoTBench will be accompanied by several tools to make it practical and easy to use allowing researchers to perform automated and reproducible execution of the benchmark, both on real testbeds (§IV-A) and in simulation (§IV-B).

Regardless of whether results are obtained experimentally or in simulation, profiles and metrics should be permanently stored in a database linked to a centralized public repository. The latter should contain all protocols and implementations that have been benchmarked, to allow anyone to re-run a given experiment or simulation, and should easily allow a user to download the source code that was used in a specific evaluation, e.g., for checking used parameters. Furthermore, a Web interface should summarize, for each profile, which solutions have performed best. An example is the EWSN competition series [3], which uses a leaderboard filtering results depending

on the selected profile, environmental characteristics, and the like.

A. Implementation on Testbeds

The low-power wireless community commonly makes use of testbed facilities to realistically evaluate and compare the performance of newly-proposed protocols to the state of the art. Large-scale public testbeds such as MoteLab, Kansei, TWIST, and—more recently—Indriya [8], FlockLab [20], and FIT IoT-Lab [1] have often been used to define an ad-hoc evaluation scenario, extract the metrics of interest from the collected serial logs, and draw conclusions about the performance of a given solution. Some of these testbeds offer valuable features such as power profiling, fine-grained GPIO tracing, and controlled mobility of nodes [21], but do not embed a number of key properties that are needed to support the benchmark proposed in (§III). We discuss next a list of these key properties and briefly sketch how they could be implemented in existing testbeds.

Extracting performance metrics. To evaluate and compare different solutions, testbeds should offer the ability to extract a minimum set of key performance metrics, such as packet delivery, end-to-end packet latency, and energy consumption. The simplest approach is to have the testbed extracting such metrics automatically by interacting with the benchmarked firmware. This, however, relies on the correctness of the firmware and has the disadvantage of not being indisputable (e.g., if the software to be benchmarked has to compute itself the energy consumption using software-based energy estimation methods [10]). The quantification of performance metrics should hence ideally be actively measured by the testbed infrastructure in an unobtrusive way. A few testbeds are capable of performing a non-intrusive fine-grained measurement of power consumption [1], [5] and timing information [21], [26]. Testbeds used as benchmarking reference can for example embed these capabilities and make use of GPIO tracing to detect the reception of packets [26] in an nearly unobtrusive way.

Feeding input parameters. To create an instance of a given profile, the testbed needs the ability to configure a number of system parameters such as the number of nodes and their topology. One could either specify offline a set of hard-coded scenarios that belong to a given benchmark profile or let the testbed automatically compute this information from an accurately derived connectivity map. Experiment parameters such as the traffic load can be “passed” to the benchmarked firmware through external scripts. For example, one can specify a list of commands passed via serial to the firmware such as “S@13@27” instructing a node to send a message to node 13 with a payload of 27 bytes. Furthermore, the testbed should be able to handle a number of experiment parameters such as the necessary number of repetitions and automatically schedule experiments accordingly.

Observing external metrics. A testbed should also have the ability to observe external metrics for each experiment. An example is the observation of the amount of external interference in the surrounding of the wireless nodes. A testbed

could make use of a few dedicated devices to scan the relevant frequencies [22] or use existing supporting tools [15]. An open research challenge is, however, how to discern the unwanted traffic from the one caused by the running experiment. Based on these, a testbed can then either autonomously refute and re-run the experiment, for example, in case the measured interference is higher than a given threshold, or automatically associate an experiment to the correct profile. Another example is the observation of people’s presence in the surroundings of the wireless nodes. Testbeds that embed cameras for monitoring the movement of robots [16] could be used for this purpose by using suitable human motion recognition algorithms.

Controlling the environment. Testbeds used to benchmark low-power wireless systems should ideally also offer the ability to control the environmental conditions to study their impact on protocol performance. For example, facilities based on off-the-shelf IEEE 802.15.4 radios can make use of Jam-Lab [4] to generate specific interference patterns over time. Towards this goal, the testbed may offer a pre-defined set of jamming profiles that can be chosen when performing an experiment, and also a way to limit the number of additional sources of interference present in the background. Similarly, specialized testbeds allowing to control the on-board temperature of nodes such as TempLab [2], could offer a pre-defined set of temperature profiles for a given set of nodes.

B. Implementation in Simulation

Many different simulators have been developed for evaluating low-power wireless protocols. A simulator brings several advantages when implementing benchmarks: the evaluations are perfectly repeatable, easy to compare against each other and easy to setup. Different benchmarks can be easily described in setup descriptions, which can be re-used by other developers. Furthermore, the evaluation is typically fast, cost-efficient and can be completely automated. However, simulations have been also criticized in the community for not being realistic enough. While higher-layer protocols and their effects can be easily implemented, other properties such as energy consumption, wireless propagation and interference, hardware faults or delays, are still hard to capture or require a complex implementation. Recently, progress has been made to address these challenges in simulation and modeling. In this section, we give an overview of the state of the art and identify some areas which still need to be addressed. In general, simulations are complementary to testbed evaluation and can be used to enable cross-validation.

Existing simulation environments. TOSSIM [19] for TinyOS pioneered many ideas in simulation of WSN applications. Cooja [23], a network simulator that supports both node emulation (only for MSP430 and AVR platforms), and native execution of Contiki applications [18], is widely used by the research community. With emulated nodes, Cooja directly runs a compiled firmware, which makes the transfer from simulation to reality easy. It also offers a number of good simulation models, e.g. for interference. Other options include OMNeT++¹, which is a general-purpose network simulator. It

¹www.omnetp.org

hosts models for various applications and the INET Framework offers a complete networking stack, including energy consumption and wireless propagation. However, it does not offer automatic transfer from simulation to real systems and code needs to be re-implemented. NS2, and its successor NS3, include sophisticated models but is not in low-power wireless networking community.

Required simulation models for benchmarking. The set of required models for simulating low power wireless systems can be separated coarsely into three groups: environmental, hardware and software models. While the software models can be easily simulated (in fact, their implementations are often identical to real systems), the environmental and hardware models pose a great challenge, both in terms of designing and implementing the models, and in using them correctly. The three groups of models are also interconnected.

a) Environment models: In these models, we strive to mimic the properties of our environment, such as wireless propagation, events (e.g., fire, movement), temperature, and humidity. As a first step towards benchmarking low-power wireless systems, we consider *wireless propagation and interference* the most important model. It describes how the waves propagate in the environment and captures properties of this propagation such as spreading, fading, attenuation, Doppler effect, and shadowing [24]. The main challenge may not be the model itself, but describing the propagation environment in the needed level of detail, including people moving around, furniture, and open/closed windows. There exist sophisticated models for wireless propagation, such as ray tracing [29] or radio irregularity model [17]. Another option is trace-based radio propagation models [14] as done in TOSSIM, for example. These models could be used in conjunction with packet reception models for synchronous transmissions [27] to accurately simulate a class of state-of-the-art protocols.

b) Hardware models: These include simulating hardware behaviour, such as delays, interrupts and errors. They also include models for power consumption and batteries. It is a rather underrepresented topic, which needs much more attention. In OMNeT++, there exists a framework for power modelling, which is still under active development [12], [13]. A lifetime estimator able to model non-linear properties of batteries was also proposed for Cooja [9].

c) Software models: These include implementations of protocols, applications and other software elements. Cooja is able to run any Contiki application (native nodes), or arbitrary firmwares (emulated nodes). In OMNeT++/INET, some implementations of IEEE standards are available, e.g. 802.15.4 or 802.11. Traffic and application models are abundant. The situation is similar for ns-2/ns-3.

In summary, the simulation environment suffers from exactly the opposite problems compared to testbeds. Extracting performance metrics, feeding input parameters, observing external metrics and controlling the environment are perfectly available in simulation, but new or consolidated models are required for accuracy and realism.

V. EARLY INITIATIVES AND ADOPTION

As a first concrete work item, we intend to evaluate a selection of multi-hop mesh routing protocols against initial, simple benchmark profiles. This will present a number of benefits: *i)* gain initial experience on building benchmark profiles, *ii)* provide data relevant to reproducibility analysis, and *iii)* raise awareness about the benchmark.

- 1) *Simple profiles:* initially, we may consider two scenarios: *i)* periodic data collection and *ii)* aperiodic any-to-any, two-way communication. Both will have well-defined input parameters and output metrics, such as packet delivery rate, end-to-end packet latency, and energy consumption. To let any interested party run their protocol, we will provide access to one reference testbed (and specify how to reproduce its infrastructure elsewhere), and define interfaces for metric computation. For instance, when it comes to measuring end-to-end packet latency, one only needs to define how packet transmission and reception events are signaled.
- 2) *Proof-of-concept testbed train:* as a second step, we aim to fully automate benchmarking and run the complete set of experiments (combination of protocol and profile) periodically, e.g., weekly. We will, in addition, consider capturing external observed metrics, such as the ambient noise level during experiments. This will enable us to study the reproducibility of experiments in non-deterministic environments, and as testbeds evolve.
- 3) *Community engagement:* finally, through publishing all results online, and encouraging the community to submit their own protocol, we will raise awareness about the initiative and receive early feedback. We believe this to be crucial towards the realization of a practical and useful benchmark.

Similar initiatives will also be instrumental in gaining adoption of the benchmark. The profiles provide ready-to-use scenarios. They will be established through a community process and will ensure that traffic patterns and application scenarios are relevant and representative. We believe that this will ease protocol evaluation as researchers do not have to identify, define, and justify their evaluation scenarios to the same degree as today. At a later stage, we will also integrate the profiles as ready to use tools in selected testbeds.

The testbed train will provide benchmark results for established protocols under above profiles. For example, it will ensure that reference implementations are configured correctly for each application scenario. Thus, a researcher does not need to evaluate and configure the protocols that a researcher wants to compare their work against, which is often a complex and time consuming process. In our experience, this testbed train will reduce the time needed for protocol evaluation. Moreover, we expect it to increase the credibility of the evaluation.

As a second concrete work item, we have started to extend the D-Cube testbed infrastructure used during the EWSN dependability competition series in an attempt to make it a full-fledged benchmarking infrastructure for low-power wireless systems [25]. Currently, D-Cube already embeds the ability of (i) extracting performance metrics; (ii) feeding input pa-

rameters such as network density, traffic load, traffic pattern, duration of a run, and number of repetitions; as well as (iii) controlling the harshness of the RF environment. Similar initiatives will shed light on the requirements of a full-fledged benchmarking infrastructure for low-power wireless systems and are also instrumental to gain adoption of the benchmark.

VI. CONCLUSIONS AND OUTLOOK

The roadmap of the proposed benchmark initiative must culminate in a set of materialized concepts with direct impact to the relevant stakeholders, including industry and academia. The most direct application is the adoption of the benchmark in the certification process chain, either by already existing certification companies or through a new entity devoted to that. In any of the cases, certification entities may exploit the benchmark procedures and results to complement or extend certification processes in order to rank the evaluated technology under certain configurations. To be successful, this initiative should seek to co-exist and complement existing industry-driven IoT benchmarks [11].

This approach may benefit the adopting entities as the performance of the technology will be fully comparable to others. At the same time, vendors would use such a reference to center their improvements or develop new products filling the technical limitations identified through the benchmark evaluation as is happening today in other verticals, e.g., CPU/processor benchmarks. Yet, the benchmark can be used by particular markets to establish entry barriers for competitors or substitute technologies. As a public reference, the benchmark can limit the required lower limits of performance for a given application domain.

Academia, in turn, will benefit from the benchmark metrics and scenarios since the benchmark results will draw the direction of what is more relevant and what are the real issues for a particular technology. This will favor the progress of research in relevant directions with direct application to adopting markets.

We believe in keeping the initiative open, enabling the benchmark initiative to progress without the influence or bias of any particular stakeholder, maintaining results and benchmark specifications away from commercial interests and ensuring the reproducibility and correctness of those publicly reported results. The benchmark community must ensure the support to open tools when possible, eliminating entry barriers to those stakeholders with promising technologies but limited resources and opportunities.

REFERENCES

- [1] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and T. Watteyne. FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed. In *Proceedings of the 2nd World Forum on the Internet of Things (WF-IoT)*, 2015.
- [2] C. A. Boano et al. TempLab: A Testbed Infrastructure to Study the Impact of Temperature on Wireless Sensor Networks. In *Proc. of the 13th IPSN Conf.*, 2014.
- [3] C. A. Boano, M. Schuß, and K. Römer. EWSN Dependability Competition: Experiences and Lessons Learned. In *IEEE Internet of Things Newsletter*, 2017.
- [4] C. A. Boano, T. Voigt, C. Noda, K. Römer, and M. A. Zúñiga. JamLab: Augmenting Sensor Testbeds with Realistic and Controlled Interference Generation. In *Proceedings of the 10th IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2011.
- [5] S. Bouckaert, W. Vandenberghe, B. Jooris, I. Moerman, and P. Demeester. The w-iLab.t Testbed. In *Proceedings of the 6th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, 2010.
- [6] Computer Vision Datasets. <http://www.cvpapers.com/datasets.html>. Accessed: 22 February 2018.
- [7] Database Benchmarks. <https://github.com/benstopford/awesome-db-benchmarks/blob/master/README.md>. Accessed: 22 February 2018.
- [8] M. Doddavenkatappa, M. Chan, and A. Ananda. Indriya: A Low-Cost, 3D Wireless Sensor Network Testbed. In *Proceedings of the 7th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, 2011.
- [9] W. Dron, S. Duquenooy, T. Voigt, K. Hachicha, and P. Garda. An Emulation-based Method for Lifetime Estimation of Wireless Sensor Networks. In *Proc. of the 10th DCOSS Conf.*, 2014.
- [10] A. Dunkels, F. Österlind, N. Tsiftes, and Z. He. Software-based On-line Energy Estimation for Sensor Nodes. In *Proceedings of the 4th International Workshop on Embedded Networked Sensors (EmNetS)*, 2007.
- [11] EEMBC IoTMark. <https://www.eembc.org/iot-connect/about.php>. Accessed: 14 March 2018.
- [12] L. M. Feeney, C. Rohner, and P. Gunningberg. Demo: Making Batteries a First Class Element in the Design and Evaluation of Embedded Wireless Systems. In *Proc. of the 14th EWSN Conf.*, 2017.
- [13] L. M. Feeney and D. Willkomm. Energy Framework: an extensible framework for simulating battery consumption in wireless networks. In *Proc. of the 3rd SIMUTOOLS Conf.*, 2010.
- [14] K. Garg, A. Förster, D. Puccinelli, and S. Giordano. A TinyOS Based Tool for Gathering Real-World Wireless Traces. In *Proc. of the 6th ACM WinTECH Workshop*, 2011.
- [15] T. Istomin, R. Marfievici, A. L. Murphy, and G. P. Picco. Trident: In-field connectivity assessment for wireless sensor networks. In *Proceedings of the 6th Extreme Conference on Communication and Computing (ExtremeCom)*, 2014.
- [16] D. Johnson et al. Mobile Emulab: A Robotic Wireless and Sensor Network Testbed. In *Proc. of the 25th IEEE INFOCOM Conf.*, 2006.
- [17] B. Khalilov, A. Förster, and A. Udugama. Radio Irregularity Model in OMNeT++. In *Proc. of the 4th OMNeT++ Community Summit*, 2017.
- [18] H.-S. Kim, J. Ko, D. E. Culler, and J. Paek. Challenging the ipv6 routing protocol for low-power and lossy networks (rpl): A survey. *IEEE Communications Surveys & Tutorials*, 19(4), 2017.
- [19] P. Levis et al. TOSSIM: accurate and scalable simulation of entire TinyOS applications. In *Proc. of the 1st ACM SenSys Conf.*, 2003.
- [20] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel. FlockLab: A Testbed for Distributed, Synchronized Tracing and Profiling of Wireless Embedded Systems. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks (IPSN)*, 2013.
- [21] R. Lim, B. Maag, B. Dissler, J. Beutel, and L. Thiele. TraceLab: A Testbed for Fine-Grained Tracing of Time Sensitive Behavior in Wireless Sensor Networks. In *Proc. of the 10th IEEE SenseApp Workshop*, 2015.
- [22] C. Noda et al. Quantifying the Channel Quality for Interference-Aware Wireless Sensor Networks. *ACM SIGBED Review*, 8(4), 2011.
- [23] F. Österlind et al. Cross-level sensor network simulation with COOJA. In *Proc. of the 1st IEEE SenseApp Workshop*, 2006.
- [24] T. S. Rappaport. *Wireless Communications*. Prentice Hall, Upper Saddle River, NJ, 2001.
- [25] M. Schuß, C. A. Boano, and K. Römer. Moving Beyond Competitions: Extending D-Cube to Seamlessly Benchmark Low-Power Wireless Systems. In *Proc. of the 1st CPSBench Workshop*, 2018.
- [26] M. Schuß et al. A Competition to Push the Dependability of Low-Power Wireless Protocols to the Edge. In *Proceedings of the 14th International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2017.
- [27] M. Wilhelm, V. Lenders, and J. B. Schmitt. On the reception of concurrent transmissions in wireless sensor networks. *IEEE Trans. on Wireless Commun.*, 13(12), 2014.
- [28] YCB Benchmarks. <http://www.ycbbenchmarks.com/>. Accessed: 13 March 2018.
- [29] Z. Yun and M. F. Iskander. Ray Tracing for Radio Propagation Modeling: Principles and Applications. *IEEE Access*, 3, 2015.