Routing Behavior across WSN Simulators: the AODV Case Study

Ivan Minakov and Roberto Passerone
Dipartimento di Ingegneria e Scienza dell'Informazione
University of Trento, Trento, Italy
ivan.minakov,roberto.passerone@unitn.it

Alessandra Rizzardi and Sabrina Sicari
Dipartimento di Scienze Teoriche e Applicate
Università degli Studi dell'Insubria, Varese, Italy
alessandra.rizzardi,sabrina.sicari@uninsubria.it

***Abstract*—The continuous interest in Wireless Sensor Networks (WSN) has led to the development of several applications, from traditional monitoring, to cooperative and distributed control and management systems, to automated industrial machinery and logistics. The design and optimization of specialized WSN platforms and communication protocols typically relies on simulation tools, which have been designed to explore and validate WSN systems before actual implementation and real world deployment. In this paper, we evaluate the performance and the accuracy of mainstream open source simulation tools for WSNs on a realistic multi-hop data passing benchmark which makes use of the Ad-hoc On Demand Distance Vector Routing (AODV) protocol. The simulation results are then compared against measurements on a physical prototype. Our experiments show that the tools produce equivalent and consistent results from a functional point of view. However, their ability to model details of the execution platform and of the communication channel may significantly impact the run-time simulation performance and the accuracy of the simulation results.**

I. Introduction

The use of Wireless Sensor Networks (WSN) has steadily increased in the last years, from the traditional monitoring applications, to becoming the adopted infrastructure for communication in buildings, open environments, smart grids, and the industrial and factory floor [1]. Despite the many research activities pursued at several academic and industrial centers in this domain, many aspects related to architectures, communication protocols and software layers are not yet well defined and standardized [2]. In contrast to traditional networks (e.g., TCP/IP), there are no common design patterns, standard APIs and hardware solutions to be applied for all possible applications. For instance, it is often the case that a sensor network system is designed essentially from the ground up, a situation that in turn requires an individual design approach in order to satisfy a range of application specific demands, especially regarding the communication strategies [3]. In addition, the wide choice of available non-standard MAC and routing protocols complicates the design process even more, making it difficult to choose the right solution that best fits a particular system [4]. As a consequence, it is essential to be able to explore and validate the various aspects of a complex design space before proceeding to the actual implementation and real world deployment.

978-1-5090-2339-4/16/$31.00

One of the most widely used approach for the evaluation and design exploration of WSN systems is simulation. Simulation allows designers to automatically estimate the parameters of a system, and can be employed when a prototype implementation is either unavailable or impractical [5]. Simulation tools offer a diversified set of facilities to model, examine and test aspects such as application algorithms, communication protocols, cooperative network behavior, and power saving techniques.

Each simulation tool is designed for a specific purpose and has a different level of accuracy and suitability depending on the target application. Therefore, it is essential to be aware of the strengths and weaknesses of the available simulators, in order to use them properly according to the purpose they were designed for. Many studies and surveys on available WSN simulation environments have been recently presented in the literature [6], [7], [8]. These provide a general panoramic view on the state of the art of WSN simulation tool-kits. However, only a few of the published papers exhibit practical studies and real test cases to explore and compare the accuracy and the usability of the various tools [9], [10], [11].

In this paper, we present a practical comparative study of a number of recent open source simulation tools for WSN systems. We extend our previous work [12] to assess different mainstream simulation environments with respect to the general usability, run-time performance and model scalability, and network model accuracy in terms of both network throughput and latency using a realistic MAC and routing protocol layer. We study the recent releases of the Castalia [13], MiXiM [14], PASES [15], WSNet [16] and COOJA [17] simulators, due to their popularity in the WSN simulation area, their completeness and their level of current development support. All the evaluated metrics mentioned above are obtained by means of a benchmark application which was implemented natively and equally in each of the studied tool-kits. The test case is a multi-hopping data passing scenario where, in particular, we analyze the performance of the Ad-hoc On Demand Distance Vector Routing (AODV) protocol [18]. In addition to the simulators, we compare the results to actual measurements obtained from the execution of the same application running on a properly configured deployment.

This paper is structured as follows. In Section II we review related work, considering various flavors of WSN simulators,

and overview related reports and practical studies of various simulation tools. Section III presents our evaluation criteria and metrics, as well as the details of our test case application. Information about the design process and the challenges of implementing a set of equivalent benchmark models in different tools and in a real deployment are considered in Section IV. Section V presents and discusses the obtained results. Finally, some conclusions are drawn in Section VI.

II. Related work

In this section we provide a broad overview of the available simulation environments for both generic networked and WSN applications, and we review the published studies on routing protocols, especially regarding the AODV algorithm.

A. Simulation environments

The whole set of available simulators for WSNs might be divided into three categories: the *general purpose* (generic) network simulators, the *network oriented* and the *sensor node oriented* frameworks, specially designed for simulation of WSN applications.

The generic network simulators are intended to model conventional networks such as the IEEE 802 family of LAN/MAN systems. Such a category include: (i) NS-2 (Network Simulator 2) [19], which is one of the most popular general purpose open source network simulators; it provides a broad variety of simulation models for widely used IP network protocols (i.e., TCP/IP, routing and multi-casting protocols for conventional wired and wireless networks); (ii) NS-3 tool (Network Simulator 3) [20], which includes models for the most popular protocols including socket API, TCP/IP, IPv6, MANET routing, IEEE802.11, WiMAX, etc; (iii) OMNet++ [21], which provides a deep analysis of network activities at the packet layer, by means of modules and channels to implement and connect simulation components.

The WSN network oriented tools focus on the network aspects of WSN systems and typically offer highly realistic and accurate models of the communication infrastructure. The most related to our work are: (i) Castalia [13], which is based on the OMNet++ framework; it includes accurate radio physical (PHY) and communication models, as well as customizable models of the most popular MAC and routing protocols for WSN applications; (ii) MiXiM (Mixed Simulator) [14], also based on OMNet++, provides detailed models of the communication channel and the radio PHY layer, with a high level of customization in terms of modulation types, sensitivity, output signal power, radio hardware operating states with power and timing parameters; (iii) PAWiS [22], built on top of the OMNet++ framework, allows to model both software components (e.g., software tasks, application, routing and MAC protocols) and hardware components (e.g., CPU, timers, Analog-to-Digital Converter, radio transceiver): (iv) WSNet [16], which offers a wide range of radio medium models including a basic ideal physical layer with no interference, no path-loss and a fix radio range, multiple frequencies, complex antenna radiation patterns, etc; (v) PASES (Power Aware Simulator for Embedded Systems) [15], [23], initially designed for executing accurate power consumption estimation for WSN hardware platforms, uses a flexible multilayer architecture which allows users to specify and assemble models of HW/SW platforms from a set of predefined components including CPU, timers, ADCs, Flash, USART, radios and other models; its communication channel is relatively simpler than the ones available in Castalia and MiXiM; (vi) Sense [24] implements each sensor node as a collection of components connected via unified input/output ports, and includes a wide a range of MAC and routing protocols.

Despite the differences among the tool-kits in terms of language, usability, and analyzed metrics, these environments also share sufficient similarities, such as simulation input, available models for the wireless channel, physical layer and MAC level protocols, and so on, which make them amenable to a performance and accuracy comparison.

Finally, as regards the *sensor node simulators* or *emulators*, they provide simple lightweight communication models. These tools are generally intended to validate and test platform specific software on top of the virtual model of the target hardware, thus they are often tied to a particular hardware architecture. Such a category includes: (i) TOSSIM, which is included in the TinyOS [25] framework; it provides a high level of scalability and execution speed for networks with a large number of sensor nodes, but it does not capture low-level details of timing and interrupts, which can be important for precise time-power analysis; moreover, the TOSSIM simulation model is not extensible and it supports only the Micaz hardware platform model; (ii) AVRORA [26], which provides high scalability, like TOSSIM, but supports only AVR MCU cores; (iii) COOJA/MSPSiM [17], which is a simulation framework for the Contiki [27] sensor node operating system; it provides support for the MSP430 microcontroller.

B. Routing protocols for WSN and available studies

The routing layer is an essential part of a communication stack in the majority of networked systems, including, in particular, conventional networks and WSNs. This layer operates on top of the MAC layer and is responsible for establishing routes and delivering data via multiple gates or hops from a source to a destination. Routing algorithms for WSNs differ from routing in conventional networks in several ways (e.g., the support of dynamic networks, self-configuration, reliability in hostile environments, energy efficiency, etc). Today, the design of efficient routing protocols for WSN systems is an intensive research area.

In our multi-hop scenario we consider the AODV algorithm [18], which was initially designed for Mobile Ad-hoc Networks (MANET) and then gained popularity in WSN applications. For instance, AODV is one of the "standard" routing algorithms for ZigBee networks. We should mention that we did not find any reported simulation results on this algorithm carried out in one of the tools included in our study. However, various studies on AODV were conducted in the NS-2 simulation environment [28], [29]. Perkins and Royer, the

authors of the AODV algorithm, presented an experimental study [18] carried out in the PARSEC simulator [30]. This work demonstrates the main features of the algorithm and analyzes its performance in mobile networks. The obtained results show the benefits of AODV in large-scale dynamic networks (i.e., 1000 nodes) over its predecessors on which it is partially based. Based on this work, we derived the simulation metrics we used for our routing comparative analysis in different tools. Chen et al. [28] presented NS-2 simulation results of the AODV protocol running on top of the slotted IEEE 802.15.4 MAC algorithm [31]. The authors studied the routing performance of the network with a moving sink under different MAC settings and velocity of the nodes. Simulation results included packet loss ratio, latency and energy consumption.

In recent years, a variety of WSN routing algorithms for industrial applications were designed and presented in the literature [29], [32], [33], [34]. In these cases, simulation is used to tune the algorithm parameters before deployment. In this sense, understanding the strengths and weaknesses of different simulation environments is pivotal to the selection of the most appropriate tool for design. The comparative study presented in this paper is intended to help this selection process.

III. Comparative study

In this section we present the benchmark methods and the application test case that we use to compare the AODV routing protocol performance in different WSN simulation environments. We aim to uniformly implement AODV in each of the investigated tool-kits and compare its behavior, although the various tools are designed for different purposes. Therefore, in this study we strive to make the environments functionally similar and compare them by providing unified input models to produce consistent simulation set-ups, topology, application and communication models.We include five recent (January 2016) mainstream open source WSN simulation tool-kits: the latest releases of the Castalia (version 3.0), MiXiM (version 2.3), WSnet, PASES, and COOJA. Our main objective is to assess numerical performance values in different simulation environments, and compare simulation outcomes with values obtained in a real-world deployment. The following simulation metrics are collected and evaluated:

- Run-time simulation performance; it is the time taken by the tool to run a certain number of modeled units and events.
- Routing layer efficiency; it is defined as the length of the route (i.e., number of intermediate nodes - hops - through which data packets passed from the sender to the recipient), conceived as the the time to discover and establish this route.
- Network throughput; it is the average rate of successfully delivered packets over a communication channel.
- Network latency (delivery delay); it is defined as the average time delay between sending data frames from the application layer of the source node to its reception at the application layer of the destination node. Along with network throughput, packet delivery depends on a number of parameters such as the application duty cycle, the MAC protocol implementation, the RF channel (e.g., obstacles) and so on.

In the following sections we provide a further description from the application scenario to the routing, MAC, PHY and RF models.

A. Application scenario

The test case scenario is designed to evaluate and compare communication stack and routing protocol performance in the multi-hop communication model for all the simulation tools included in this study. The modeled network topology is distributed on a 40 by 60 meters indoor space, composed by corridors at a building floor, according to the real-world reference deployment shown in Figure 1 [35], [36]. This deployment presents a centralized network topology for indoor environmental monitoring. Currently it consists of 25 wireless nodes continuously measuring various indoor conditions including temperature, light, humidity, vibration and electrical load. In the presented scenario we study network performance for various numbers of traffic generator nodes (senders) that forward data packets to the sink. Senders periodically broadcast packets with a constant bit rate (CBR) at 1 packet per minute. In our study we evaluate 1, 2, 5, 10 and 24 CBR nodes in the network.

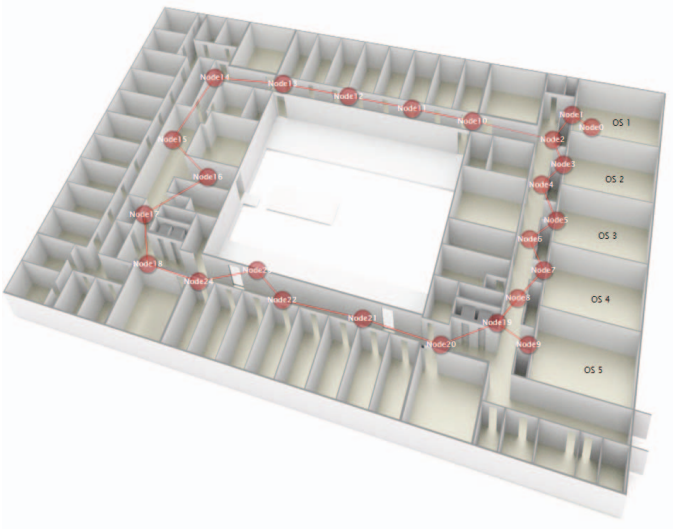

Fig. 1: Network topology consisting of 25 nodes

B. Routing model

The multi-hop communication including route discovery, path maintenance and message passing from senders to the sink is handled by the AODV routing algorithm. This is a reactive routing protocol that uses an on-demand approach to find and establish routes among nodes. AODV initiates route discovery whenever a data transfer is required, thus avoiding the use of redundant memory to maintain unused routes in routing tables. This algorithm tries to find the shortest

possible path that does not contain loops. AODV achieves better performance when nodes have dynamic configurations. However, in our study we do not consider nodes mobility as well as power consumption aspects of the AODV protocol. AODV consists of two routing phases called *discovery* and *maintenance*. The discovery phase establishes the route from source to destination based on query and reply procedures, while intermediate nodes create routing table entries along the path, as shown in Figure 2. Route discovery is initiated when a source node (node 0 in Figure 2) wants to find a route to a new destination (node 3 in Figure 2) or when the lifetime of an existing route to a destination has expired. The process is initiated by broadcasting a Route Request Message (*RREQ*) to the neighbor nodes. The *RREQ* message contains several important fields: the source, the destination, the lifespan of the message (*TTL*) and a sequence number which acts as a unique ID. The message keeps getting rebroadcast until either a route is discovered or its *TTL* expires. If an intermediate node knows the route to the destination, or if the destination node is reached, a Route Reply Message (*RREP*) packet is sent back to the source along the path.

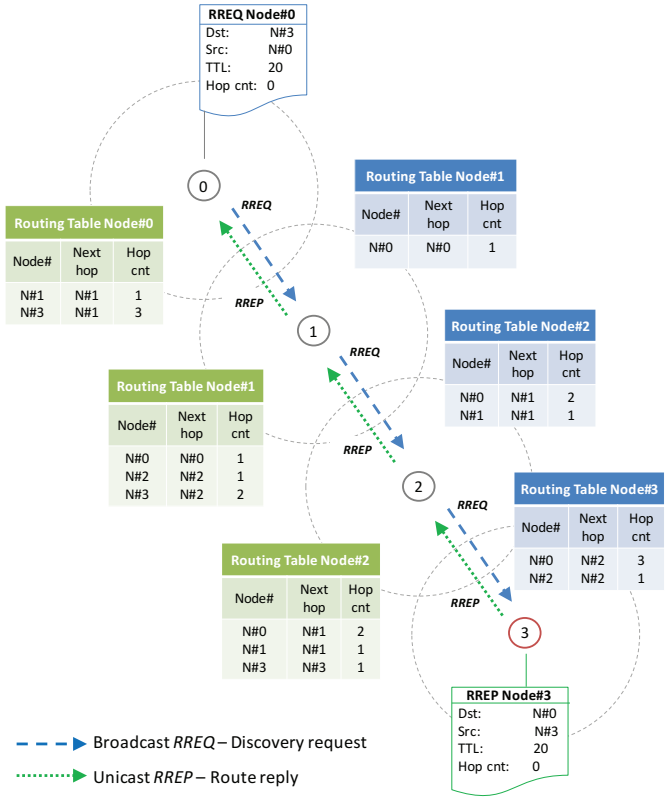

Fig. 2: Route discovery in AODV

The state of an established route includes an expiration timer, which is used by AODV to invalidate a route after it has not been used after a certain period of time. Optionally, AODV utilizes *HELLO* messages for path maintenance and discovery of its closest neighbors. Each node periodically sends this message to update and notify neighbors about its state information. The *HELLO* message contains fields such as the known active routes (routing table), the geographical location, and the residual battery level.

C. MAC model

All network transmissions in our scenario rely on an un-slotted IEEE 802.15.4 CSMA/CA algorithm [31] (see Figure 3). According to the IEEE standard, each network device operates with a set of variables which define the boundaries for carrier sense and back-off operations. *NB* is the number of times the CSMA/CA algorithm is required to back-off while attempting the current transmission; this value is initialized to zero before each new transmission attempt. *BE* is the back-off exponent, which is related to how many back-off periods a device shall wait before attempting to assess the availability of the channel [31]. Before each transmission, *BE* is initialized to

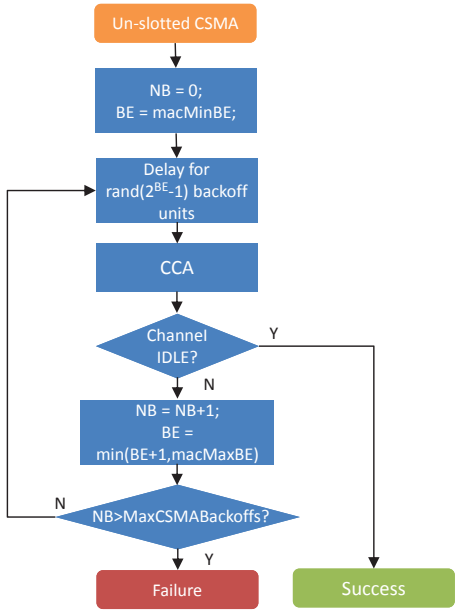

Fig. 3: Un-slotted IEEE 802.15.4 CSMA/CA algorithm

a defined value *macMinBE*. When a device starts to transmit data, it initializes *NB* and *BE*, and waits for a random number of back-off periods defined by *BE*. The duration of the back-off period (backoff unit) is defined by a constant value held in a variable *aUnitBackoffPeriod*. Subsequently, after back-off, the node immediately performs a Clear Channel Assessment (CCA) to check whether the medium is idle. If so, the data is transmitted, otherwise the device increases both *BE* (up to a maximum value) and *NB*, and backs off once again. This procedure is repeated until either CCA reports idle and the packet is transmitted, or *NB* exceeds the maximum allowed number *(MaxCSMABackoffs)* of CSMA/CA back-offs. In the latter case, a MAC error occurs and the current packet is discarded from the node's transmission queue.

The parameters for the MAC model have been set to the default values specified in the IEEE 802.15.4 standard, namely *MaxCSMABackoffs* was set to 4, the *BE* boundaries values

macMinBE and *macMaxBE* were set to 3 and 5 respectively, and *aUnitBackoffPeriod* was equal 0.96 milliseconds.

For the sake of simplicity, the acknowledgments and any handshaking algorithms (such as RTS-CTS) are not considered in our model. For all transmissions, the packet size is fixed to 64 bytes (16 bytes for the MAC header and the rest for the data payload). The sink node collects statistics on the total number of successfully received packets and computes the mean delivery latency by analyzing the timestamp included into each packet.

D. Radio channel model

Along with the application, routing and the MAC models, the settings of the radio propagation model have been carefully chosen in order to provide identical communication environments among the studied tool-kits. We consider a propagation model based on the log-distance path loss equation [37], which is available in all the tools out of the box:

$$PL_{dB}(d) = PL_{dB}(d_0) + 10\eta \log_{10}\left(\frac{d}{d_0}\right) + X_{\sigma,dB} \quad (1)$$

where d_0 is the reference distance, η is the path loss exponent that defines the rate at which the signal decays with respect to the distance d, and $X_{\sigma,dB}$ is a zero-mean Gaussian random variable (in dB) with standard deviation σ. The last term represents the signal shadowing effect that accounts for dependencies of all environmental factors, such as static and mobile obstacles and signal reflections.

A simple interference model is chosen in all tools to handle signal collisions. In this model, packet collisions happen and transmissions fail (packets are marked as invalid) every time two or more nodes are concurrently transmitting within the receiver range. In order to make simulated communication model more realistic, we include properties of a real RF subsystem utilized in the reference network, by assuming a NXP JN5148 radio model for all the simulated nodes. The RF systems are set to operate at a radio frequency of 2.4 GHz with a data rate of 250 kbps, -3 dBm output power and -85 dBm signal sensitivity. the network density and path length appear to be a function of the network size, placement area and nodes communication ranges. Table I summarizes the global settings for the benchmark test case presented in this paper.

TABLE I: Case study settings summary

Network area, m x m	40×60
Traffic type/rate (pkt/min)	CBR/1
Network size (N)	25
Number of senders	1, 2, 5, 10, 24
Data packet size (bytes)	64
Routing	AODV
MAC	IEEE802.15.4
PHY models	NXP JN5148
RF output power/ receiver sensitivity	-3dBm/-85dBm
Communication channel model	log-normal shadowing $\eta = 4.0$, $\sigma = 20$
Sim time, sec	3600

IV. Implementation

All the tools in our study have many similarities regarding architecture, design principles and simulation configuration. They have an object-oriented architecture based on an event-driven simulator engine implemented in C++ or, for COOJA, in C and Java. Simulation configuration and network set-ups in all environments are made by means of external configuration and scripting files.

Castalia and MiXiM utilize OMNet++ based initialization files (.ini and .ned formats). Inside the .ned files, the topology of simulation modules can be defined as an interconnection of simple components linked through channels and interfaces. Additionally, .ned files hold default values for the module external variables and constants. The .ini format files are used to configure simulation scenarios. They typically include the number of simulated nodes, node assigned locations, MAC modules, simulation time and so on. Castalia also includes a set of plain text files to configure power-timing properties of the RF section, while MiXiM uses .ned files for this purpose. Both WSNet and PASES employ XML configuration files for the network and simulation set-ups. Additionally, PASES offers Python scripting facilities for MAC and application design as well as external Python files to hold energy-timing values for hardware models. COOJA supports the creation of the desired simulations by first indicating the radio medium to be used, and then by instantiating the nodes and by compiling their code, which was previously written in the C language. Once the network parameters have been defined (e.g., area, node number, topology), the simulation environment can be stored in a .csc file, in order to save it for future usage. Note that from the COOJA GUI, only the network configurations can be set, while the node behavior has to be defined at the Contiki code level; the file projec-conf.h contains all the defined configurations for a specific node.

With the exception of COOJA, none of the other tools studied in this paper includes an AODV model implementation out of the box. Thus, this protocol was designed from scratch and then ported and tested on each tool. For the sake of simplicity, we restricted the functionality of the original AODV [18]. Since the modeled topology and node configuration is static, we do not include the path *maintenance* phase in our model. Namely, *HELLO* messages and route expiration timeouts are not implemented. We assume that all the nodes and routes remain unchanged over simulation time. Additionally, only destination nodes are allowed to generate *RREP* packets in response to the *RREQ* during the route discovery procedure. Due to the relative simplicity of the implemented algorithm and the modular architecture of the studied tool-kits, we did not encounter any considerable challenges while porting this routing model into different tools.

Functionally equivalent version of the application, AODV and MAC algorithms were implemented on the real reference deployment. Figure 4 shows single NXP J5148-based node

utilized in the experiments. The sink node was connected to a PC in order to profile the received packets and perform the further analysis on throughput and latency. Code for all 25 nodes was written in C on top of the NXP low-level API which allows the application to access platform specific functionality, such as radio CCA, timers, interrupts, etc. Additional functionality was implemented for round-trip time estimation between sink node and each sensor node. This round-trip delay is profiled for an accurate timestamp analysis and latency estimation. The core of the MAC model,

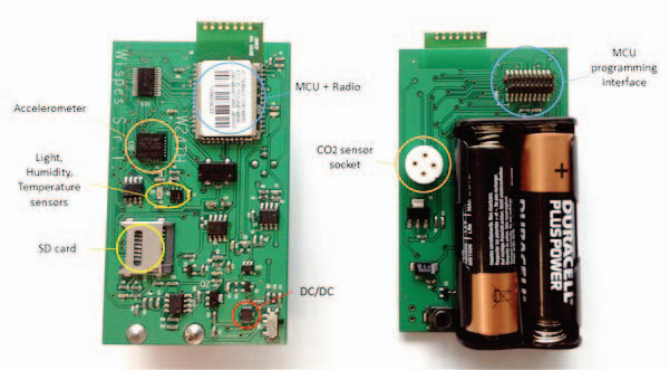

Fig. 4: NXP J5148 node

which is the CSMA algorithm (see Section III), has been implemented in the studied tools. It involves radio module CCA functionality and a single timer component in order to process backoff periods. As mentioned in Section IV, for the tools Castalia, MiXim, WSnet and PASES we selected and set up the communication channel model based on the log-normal shadowing models available in each of these tool out of the box, while for COOJA the Multi-path Ray-tracer Medium (MRM) propagation model is considered, due to its capability of setting the proper parameters to simulate the shadowing channel conditions. Further analysis revealed that the differences in simulation results were caused mostly by the differences in the implementation of communication channel models in different tools. The results of the network throughput study are reported in Section V.

V. Results

Our evaluation is based on a series of simulations executed to compare the mentioned tool-kits with respect to the metrics discussed in Section III. All simulations were conducted on a desktop PC equipped with an AMD Athlon 64 X2 5000+ 2.6 GHz with 4GB of RAM running either Ubuntu 11.10 or the Windows 7 Professional operating system. The simulations were all set up to simulate 3600 seconds of virtual operating time.

Figure 5 shows the obtained run-time performance in various tools. As expected, the run-time of all of them grows almost linearly as both the number of senders and the number of packets increase. The performance of the simulators is faster than real time and a noticeable difference among the tools emerges. In fact, the performance of PASES appears to be the worst among all the studied tool-sets. The high computation demands shown by PASES are explained by its node-oriented architecture that captures the low-level features of the underlying hardware models. That, in turn, leads to an increased number of events and simulation overhead. The MiXiM tool also exhibits relatively low run time performance. We were unable to identify the actual reason for the poor performance in MiXiM, and additional experiments would be required to find performance bottlenecks in its simulation algorithm. In contrast to the PASES and MiXiM tools, the WSNet tool-kit is found as the fastest and most scalable simulation environment among the ones compared. Castalia and COOJA show run-time results comparable to WSNet, making them also efficient tools for rapid WSN analysis.

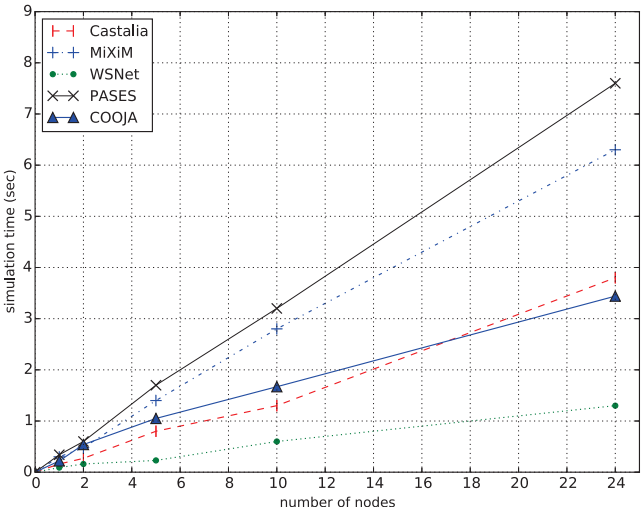

Fig. 5: Simulation performance

The AODV settings were uniformly configured as follows: Route Discovery Timeout is 1000 milliseconds, packet TTL is 20 hops, routing header is 12 bytes long, nodes start up delay is chosen randomly between 0 and 2 seconds. Simulation begins with the route discovery procedures initiated by the traffic generator nodes. Because the network configuration is static and the route expiration timeout is disabled, this procedure happens only once for each sender. Figure 6 shows the route acquisition latency obtained in different tools. The established paths are the same in all the tools, but the time taken to explore these routes varies slightly among them. COOJA demonstrate the lowest delay with respect to the other simulators, which show similar performance (among them, WSNet shows the highest level of discovery latency), due to its more accurate channel model.

Upon successful completion of the discovery phase, the senders start to generate unicast traffic directed to the central node through the established paths. Figure 7 presents

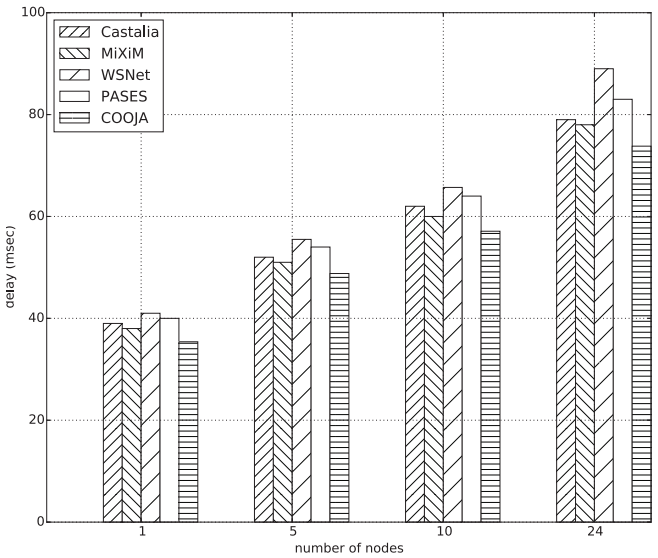

Fig. 6: AODV route discovery latency

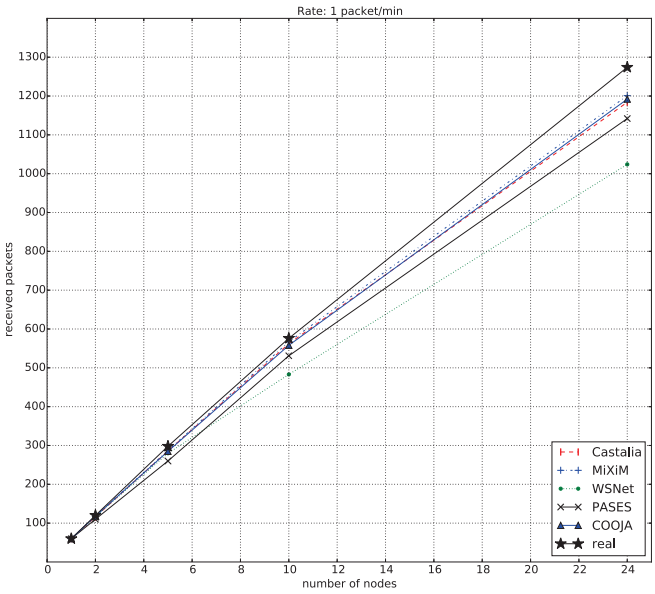

Fig. 7: Network througput

the throughput obtained for different transmission rates and network sizes. The network throughput is calculated as the total number of received packets at the sink side. The figure also presents results obtained on the reference deployment which is functionally equivalent to the modeled application.As expected, the number of successfully delivered packets in both simulations and reference deployment steadily increases with the number of CBR nodes. However, the throughput measured on the real test-bed is always higher than the throughput obtained in each of the other simulations. It can be explained by shorter AODV route paths (lower number of hops) due to the specific indoor environment (i.e., long corridors that behave like a waveguide for radio signal).

The packet delivery latency is depicted in Figure 8. The latency is calculated as an average delivery delay between the sink and sender nodes over simulation time. The delivery delay increases with the number of CBR nodes in the network due to the growing number of contending transmissions and extended CSMA backoff time. In all the cases the real test-bed demonstrates lower latency compared to the results obtained in simulations. As for throughput results, it can be explained by shorter AODV route paths and specific real-life radio phenomena such as capturing effect, which is not modeled in any of the simulation tools. All the tools exhibit predictable and highly comparable results in terms of packet delivery dynamics as well as routing layer functionality. As it was revealed before, COOJA shows the lowest run time execution efficiency with respect to the other simulators, which instead present similar performance, again due to the higher accuracy of its channel model.

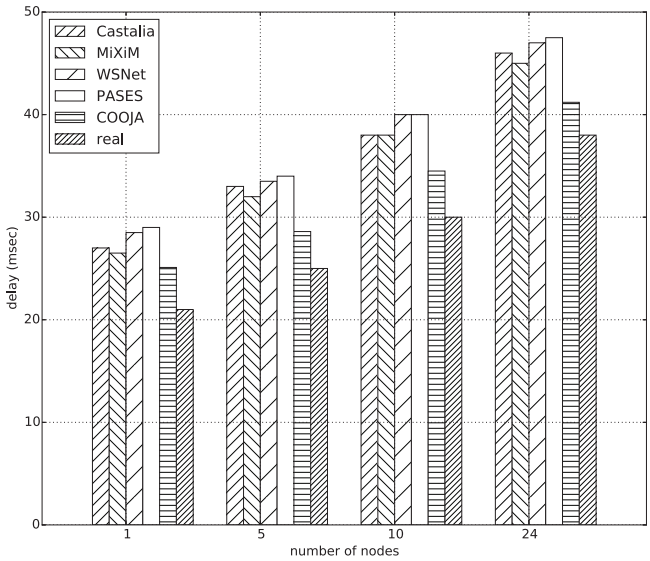

Fig. 8: AODV packet delivery latency

VI. Discussion and Conclusion

In this paper we have presented a comparative study of five open-source WSN simulation tool-kits based on a building environmental monitoring application that uses the AODV routing protocol. A multi-hop simulation scenario was designed and equally implemented in each of the tools to assess and compare them with respect to network performance, latency, throughput, and routing layer efficiency. Additionally, simulation results were compared with measured values obtained on a real test bench. Despite the differences in simulation analysis, capabilities and available component models, the results show the correctness of the benchmark methods adopted. The obtained results proved the functional equivalence of the tools and their applicability for multi-hop network modeling. At the same time, differences in the ability of the tools to model detailed node behavior greatly affect simulation performance. While these extra capabilities were not used in our test case

scenario, a detailed node description is generally essential to study aspects such as power consumption, or to accurately estimate the utilization of the node computing platform. While throughput is very consistent across the simulators, latency has a wider variability. This is due mostly to differences in the modeling of the communication channel. The differences in simulations results and measurements obtained on the real test bench are due to the limitations of the applied modeling assumptions that do not address real-life phenomena such as the waveguide effect in indoor communication environments.

REFERENCES

[1] V. Gungor and G. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *Industrial Electronics, IEEE Transactions on*, vol. 56, no. 10, pp. 4258–4265, Oct 2009.

[2] L. Mottola, G. P. Picco, M. Ceriotti, c. Gună, and A. L. Murphy, "Not all wireless sensor networks are created equal: A comparative study on tunnels," *ACM Trans. Sen. Netw.*, vol. 7, no. 2, Sep. 2010.

[3] L. Lo Bello and E. Toscano, "Coexistence issues of multiple co-located IEEE 802.15.4/ZigBee networks running on adjacent radio channels in industrial environments," *Industrial Informatics, IEEE Transactions on*, vol. 5, no. 2, pp. 157–167, May 2009.

[4] K. Langendoen and A. Meier, "Analyzing MAC protocols for low data-rate applications," *ACM Trans. Sen. Netw.*, vol. 7, no. 2, Sep. 2010.

[5] A. Davare, D. Densmore, L. Guo, R. Passerone, A. L. Sangiovanni-Vincentelli, A. Simalatsar, and Q. Zhu, "METROII: A design environment for cyber-physical systems," *ACM Transactions on Embedded Computing Systems*, vol. 12, no. 1s, pp. 49:1–49:31, March 2013.

[6] J. Haase, J. Molina, and D. Dietrich, "Power-aware system design of wireless sensor networks: Power estimation and power profiling strategies," *Industrial Informatics, IEEE Transactions on*, vol. 7, no. 4, pp. 601–613, Nov 2011.

[7] C. P. Singh, O. P. Vyas, and M. K. Tiwari, "A survey of simulation in sensor networks," in *Proceedings of the International Conference on Computational Intelligence for Modelling Control Automation*, Vienna, Austria, December 10-12 2008, pp. 867–872.

[8] M. Korkalainen, M. Sallinen, N. Kärkkäinen, and P. Tukeva, "Survey of wireless sensor networks simulation tools for demanding applications," in *Proceedings of the Fifth International Conference on Networking and Services*, Valencia, Spain, April 20-25 2009, pp. 102–106.

[9] J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus, "Comparative study of wireless network simulators," in *Proceedings of the 7^{th} International Conference on Networking*, Cancun, Mexico, April 13-18 2008.

[10] A. Timm-Giel, K. Murray, M. Becker, C. Lynch, C. Gorg, and D. Pesch, "Comparative simulations of WSN," in *Proceedings of ICT Mobile and Wireless Communications Summit*, Stockholm, Sweden, June 10-12 2008.

[11] H. Lee, A. Cerpa, and P. Levis, "Improving wireless simulation through noise modeling," in *Proceedings of the 6^{th} International Conference on Information Processing in Sensor Networks*, ser. IPSN '07, Cambridge, Massachusetts, USA, 2007, pp. 21–30.

[12] I. Minakov, R. Passerone, A. Rizzardi, and S. Sicari, "A comparative study of recent wireless sensor network simulators," *ACM Transactions on Sensor Networks*, 2016.

[13] A. Boulis, "Castalia: revealing pitfalls in designing distributed algorithms in WSN," in *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*, ser. SenSys, Sydney, Australia, November 6-9 2007, pp. 407–408.

[14] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, and S. Valentin, "Simulating wireless and mobile networks in OMNeT++ the MiXiM vision," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, ser. Simutools, Marseille, France, March 3-7 2008, pp. 71:1–71:8.

[15] I. Minakov and R. Passerone, "PASES: An energy-aware design space exploration framework for wireless sensor networks," *Journal of Systems Architecture*, vol. 59, no. 8, pp. 626–642, September 2013.

[16] C. Guillaume, F. Antoine, and E. Fleury, "Worldsens: a fast and accurate development framework for sensor network applications," in *Proceedings of the 22^{nd} ACM Symposium on Applied Computing*, Seoul, Korea, March 11-15 2007, pp. 222–226.

[17] J. Eriksson, F. Osterlind, T. Voigt, N. Finne, S. Raza, N. Tsiftes, and A. Dunkels, "Accurate power profiling of Sensornets with the COOJA MSPSim simulator," in *Proceedings of the 6^{th} IEEE Conference on on Mobile Adhoc and Sensor Systems*, Macau, China, October 12-15 2009.

[18] C. E. Perkins and E. M. Royer, "Ad-hoc on demand distance vector routing," in *Proceedings of the 2^{nd} Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 25-26 1999.

[19] I. T. Downard, "Simulating sensor networks in NS-2," Naval Research Laboratory, Network and Communication Systems Information Technology Division, Washington, DC, USA, Tech. Rep., May 31 2004.

[20] T. R. Henderson, M. Lacage, and G. F. Riley, "Network simulations with the ns-3 simulator," in *Proceedings of the Special Interest Group on Data Communications*, Seattle, WA, USA, August 17-22 2008.

[21] A. Varga, "The OMNeT++ discrete event simulation system," in *Proceedings of the 15^{th} European Simulation Multiconference*, Prague, Czech Republic, June 6-9 2001, pp. 319–324.

[22] D. Weber, J. Glaser, and S. Mahlknecht, "Discrete event simulation framework for power aware wireless sensor networks," in *Proceedings of the 5^{th} IEEE International Conference on Industrial Informatics*, Vienna, Austria, July 23-27 2007, pp. 335–340.

[23] A. Somov, I. Minakov, A. Simalatsar, G. Fontana, and R. Passerone, "A methodology for power consumption evaluation of wireless sensor networks," in *Proceedings of the 12^{st} IEEE International Conference on Emerging Technologies and Factory Automation*, Palma de Mallorca, Spain, September 22-25 2009, pp. 1–8.

[24] B. K. Szymanski and G. G. Chen, *Handbook of Dynamic System Modeling*. CRC/Taylor and Francis Publ, 2007.

[25] P. Levis, S. Madden, D. Gay, J. Polastre, R. Szewczyk, A. Woo, E. A. Brewer, and D. E. Culler, "The emergence of networking abstractions and techniques in TinyOS," in *Proceedings of the 1^{st} Symposium on Networked Systems Design and Implementation*, San Francisco, California, USA, March 29-31 2004, pp. 1–14.

[26] B. L. Titzer, D. K. Lee, and J. Palsberg, "Avrora: scalable sensor network simulation with precise timing," in *Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks*, ser. IPSN, Los Angeles, CA, USA, April 25-27 2005, pp. 477–482.

[27] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *Proceedings of the 29^{th} IEEE Conference on Local Computer Networks*, Tampa, FL, November 16-18 2004, pp. 455–462.

[28] C. Chen and J. Ma, "Simulation study of AODV performance over IEEE 802.15.4 MAC in WSN with mobile sinks," in *Proceedings of the 21^{st} International Conference on Advanced Information Networking and Applications*, Ontario, Canada, May 21-23 2007, pp. 159–164.

[29] P. Samundiswary and P. Dananjayan, "Performance analysis of energy aware ad hoc on demand distance vector routing protocol for wireless sensor networks," *International Journal of Computer Applications*, vol. 34, no. 7, pp. 53–56, November 2011.

[30] R. Bagrodia, R. A. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, and H. Y. Song, "Parssec: A parallel simulation environment for complex systems," *IEEE Computer*, vol. 31, no. 10, pp. 77–85, 1998.

[31] *IEEE 802.15.4-2006 Standard for information technology*, IEEE Standard Association, 2006.

[32] J. Niu, L. Cheng, Y. Gu, L. Shu, and S. Das, "R3E: Reliable reactive routing enhancement for wireless sensor networks," *Industrial Informatics, IEEE Transactions on*, vol. 10, no. 1, pp. 784–794, Feb 2014.

[33] F. Tang, L. Barolli, and J. Li, "A joint design for distributed stable routing and channel assignment over multihop and multiflow mobile ad hoc cognitive networks," *Industrial Informatics, IEEE Transactions on*, vol. 10, no. 2, pp. 1606–1615, May 2014.

[34] P. T. A. Quang and D.-S. Kim, "Throughput-aware routing for industrial sensor networks: Application to ISA100.11a," *Industrial Informatics, IEEE Transactions on*, vol. 10, no. 1, pp. 351–363, Feb 2014.

[35] D. Brunelli, I. Minakov, R. Passerone, and M. Rossi, "POVOMON: an ad-hoc wireless sensor network for indoor environmental monitoring," in *Proceedings of the 2014 IEEE Workshop on Environmental, Energy and Structural Monitoring Systems*, ser. EESMS14, Naples, Italy, September 17–18, 2014.

[36] ——, "Smart monitoring for sustainable and energy-efficient buildings: a case study," in *Proceedings of the 2015 IEEE Workshop on Environmental, Energy and Structural Monitoring Systems*, ser. EESMS15, Trento, Italy, July 9–10, 2015.

[37] T. S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. New Jersey: Prentice Hall PTR, 2002.