EURASIP Journal on
Wireless Communications and Networking
a SpringerOpen Journal

REVIEW Open Access

Modeling and simulation of networked low-power embedded systems: a taxonomy

Wan Du[1,2], Fabien Mieyeville[1], David Navarro[1*], Ian O'Connor[1] and Laurent Carrel[1]

Abstract

Simulation has been widely adopted for the evaluation of novel protocols or other designs for networked low-power embedded systems, especially for wireless sensor networks (WSNs). A large number of simulation tools have been developed for WSNs in the past few years. However, different tools may emphasize on different features. For example, general network simulators mainly focus on the high-level performance evaluation with certain assumptions, while some SystemC-based simulators have been developed recently to realize the hardware and software co-design of sensor node at the system level. Therefore, it is necessary to study the different modeling methodologies and to distinguish the various features of the existing WSN simulators. In this paper, we propose a taxonomy that categorizes the existing simulation tools into four groups, i.e., network simulators with node models, network simulators with node emulators, node system simulator with network models, and node emulators with network models. To demonstrate the rationality and usability of the proposed taxonomy, we use it to conduct a survey of the existing simulation tools. This study is intended to be comprehensive to cover all important simulation tools.

Keywords: Wireless sensor networks; Simulation; Modeling; Taxonomy

Review

Introduction

Wireless sensor networks (WSN) are large-scale *ad hoc* networks of resource-constrained sensor nodes that are deployed at different locations and could cooperatively monitor the physical or environmental conditions, such as temperature, vibrations, and motions. The resources on sensor nodes (e.g., energy, memory, and processing ability) are very restricted. Since the network is often expected to be self-powered and operate over periods of many months or years, the energy consumption is one of the most essential concerns during the protocol and application design. Due to the low power supply, the communication distance is normally short and the sensor data should be relayed to the gateway through multihop path. According to the unique features of WSNs, many hardware platforms of sensor nodes have been developed, such as N@L [1], MICAz [2], Telosb [3], Imote2 [4], and TinyNode [5]. These platforms possess different features. For example, compared to the others, some of them provide long communication distance, whereas some offer higher capacity of computation and memory. In addition, various protocols and standards, especially on media access control (MAC) and network layers, have been proposed like IEEE 802.15.4 [6] and Zigbee [7].

To evaluate the performances of these protocols on particular hardware platforms, simulation has been widely used [8,9]. Simulation provides a good approximation at lower cost and often in less time. For example, for one application lasting 27.8 h with a sensing interval of 10 s, the simulation time of NS-2 (open-source) is just 24 min. In addition, it also offers an easy-to-use debugging environment and a better insight of network behaviors. Lots of simulators for WSNs have been developed in the past few years. But different simulators may be designed to accomplish different target applications. For example, some simulators are intended to simulate the performance of communication protocols and some others may be designed to emulate the execution of the binary code. It is thus important to find their similarities and differences. There are already some surveys on simulation tools for WSNs [8,10]; however, the existing surveys of simulation tools mainly focus on the comparison of some particular simulators. Different to these existing works, this paper

*Correspondence: david.navarro@ec-lyon.fr
[1]Lyon Institute of Nanotechnology, University of Lyon, Lyon 69007, France
Full list of author information is available at the end of the article

emphasizes on the analysis of modeling methodologies and the various features of the current WSN simulators. An explicit classification method is developed according the features of WSN modeling and simulation.

In this paper, the modeling and simulation of WSN systems are investigated. A WSN system is mainly composed of two parts: sensor nodes and network connecting all nodes. Therefore, we need to model these two parts while evaluating the performance of this system. To model the sensor nodes, we can either use system-level simulation or instruction-level emulation. The latter refers to instruction set simulation (ISS) or operating system code emulation. For the network modeling, we need to abstract the behaviors of each node and schedule the events from different nodes. Based on an elaborate study of WSN simulations and the existing simulation tools, we proposed a taxonomy that categorizes the existing simulation tools into four categories, i.e., network simulators with node models, network simulators with node emulators, node system simulator with network models, and node emulators with network models. We analyze the main features of each type of simulators. According to the taxonomy, a comprehensive study of the existing simulation tools for WSN is conducted. Based on the results, an appropriate simulator can be selected for different applications with various requirements.

The rest of this paper is organized as follows. The 'Related works' section reviews the related works. In the 'Modeling and simulation of WSNs' section, a typical model of WSN system is provided and the requirements of WSN simulations are summarized. A taxonomy of existing WSN simulation tools is proposed in the 'Taxonomy on WSN simulation tools' section. According to this taxonomy, in the 'Survey on WSN simulation tools' section, a survey of existing WSN simulation tools is presented. We conclude this paper in the last section.

Related works

Stojmenovic [11] challenges some of the existing criticisms for simulation practices that emphasized validation aspects. He then advocates a stepwise approach which studies one variable at a time and adds more complex models gradually to obtain a full explanation for the performance of protocols.

In [8], the authors propose a general component model of the WSN system, derived form [12,13], which includes network and node models. They divided the WSN simulators into two groups: general simulation packages (e.g., NS2 and OMNET++) and specific WSN frameworks (e.g., TOSSIM and ATEMU). We further improve the general component model of the WSN system by emphasizing on the hardware components and their connections. We propose a taxonomy considering the features of different modeling methods.

Haase et al. [10] review the power estimation and power profiling strategies for WSNs. They categorize the WSN simulation tools from cycle accurate simulation to pure functional simulation into three groups: microcontroller emulators, operating system emulator, and network and system simulator. They mainly focus on the power consumption estimation. However, we consider more general methodologies of existing WSN simulation tools.

Sundani et al. [14] compare 14 simulators and discuss their advantages and disadvantages. They also use a case study to demonstrate a detailed performance comparison of NS-2 [15], TOSSIM [16], and Shawn [17]. Unlike this work, besides a comprehensive survey, we also focus on the analysis of different modeling methods. Many topology control schemes in WSNs [18], protocols in delay tolerant networks [19], routing protocols [20,21], and techniques used in green mobile networks [22] are evaluated by simulations.

Part of this work was published in the conference of ICST SIMUTools 2010 [23]. This work comprehensively extends the previous works with detailed analysis and more recent works on simulation tools for networked embedded systems. Compared with the existing survey or classification schemes for simulations in WSNs, our taxonomy is more comprehensive. It can be used by different communities of researchers working on WSNs with different requirements on simulation tools.

Modeling and simulation of WSNs

WSN mainly involves three parts: node system, network, and physical environment. A typical model of WSN system is presented in Figure 1.

In this model, the node system is composed of two parts: hardware and software. The hardware platform consists of a processing unit, RF transceiver, sensor, and battery. The software model includes an operating system, protocol stack, application software implementation, and so on. Nodes are connected to each other by the wireless network model that maintains the network topology and transfers packets among nodes. It also implements many radio frequency channel models. The environment model specifies how the physical parameters in the environment vary in both spatial and temporal sense.

By taking into account the special characteristics of WSN systems and the requirements of different WSN design fields (e.g., communication protocol design, application design, and node system design), we summarized the following six key requirements that are important to a WSN simulation framework:

- *Fidelity*. The main purpose of simulation is to model the real-world system faithfully and predict the system's behavior. For WSN, it requires accurate models of radio channels, physical environment, and

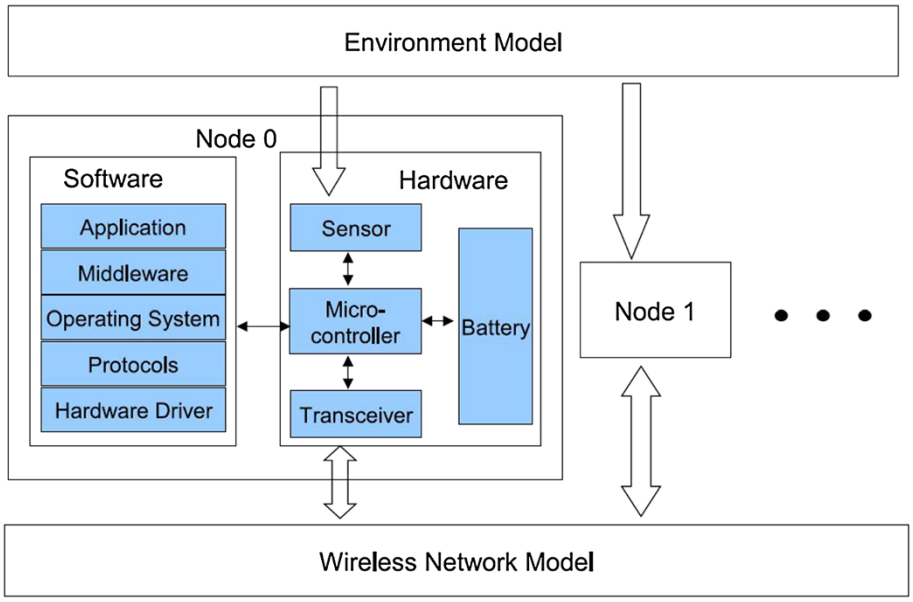

Figure 1 A typical model of WSN system.

node system. Inaccurate simulation may lead to erroneous conclusions. For example, an ideal battery model usually treats the battery as a reservoir of energy from which the energy consumption can be subtracted. However, this is not accurate as a real battery that shows non-linear discharge behavior and recovery effects. It is proved that the accuracy of battery models can affect the route fluctuations and routing overheads [24].

- *Scalability*. Because nodes are often deployed in large quantities in many WSN applications, the simulator should well support the scalability. The simulation time should be short.
- *Energy aware*. Due to the limited power supply on sensor nodes, network designers need to obtain accurate power consumption and timing figures to tune their applications before the deployment in real environments [25]. Therefore, the simulator shall be able to accurately capture the energy consumption and timing information of hardware and software (HW/SW) operations.
- *Extensibility*. It shall be easy to modify the existing modules or integrate some new ones. A careful structure with clean interfaces and high modularity allows the users to easily add or change functionality.
- *Heterogeneity support*. Many recently deployed WSN systems are heterogeneous systems, incorporating a mixture of elements with widely varying capabilities [26]. Therefore, modeling different kinds of nodes and managing the interconnections among them are necessary in WSN simulations.
- *Easy to use*. A graphical user interface (GUI) can facilitate and speed the establishment of the network topology and the composition of basic modules. It can also allow the quick visualization of the simulation results. In addition, it supports to trace and debug the simulation at real time.

There is always a trade-off between fidelity and scalability [8]. Better fidelity involves more complex and detailed modeling. However, the simulators need more time to deal with the additional detail. The simulation time may become intolerable if the number of nodes is very large in some WSN applications. Thus, the high-level abstraction is sometimes more suitable for implementing the simulation with proper complexity and little running time. The results of high-level abstraction are detailed enough to answer the design questions at early stages of design flow. For example, at the beginning of a system design, the need to quickly explore a variety of alternatives is more important than a detailed result for a specific scenario. The challenge is to identify which level of detail does not affect answers to the design questions at hand.

Taxonomy on WSN simulation tools

Simulation has been used in both node system and protocol designs to help the designer easily evaluate their new designs. Environment modeling of WSNs is still at the beginning of development. A more detailed description of environment modeling can be found in [27]. Since only few simulators have addressed environment modeling well, our taxonomy will not treat it as a determinant.

We mainly focus on the node system and network modeling.

At the beginning, node system development and protocol design are addressed by different people with different knowledge and tools. In the context of node system design, the aim is to design the nodes' hardware, to implement the software running on the hardware and to co-design the HW/SW of a single node [28,29]. The network performance of these nodes cannot be simulated in this stage. In the context of protocol design, many tools are used to model the protocols, manage the concurrency among different nodes, and simulate the throughput of the network. The protocol designers often make simple assumptions to the behavior of hardware and software, but this may be not detailed enough for some applications. For example, timing information in instruction granularity shall be considered to the fast routing lookups [30]. In addition, it is better to compress the data by processing them in a local CPU rather than transmitting the raw data to the destination node in some applications, since wireless communication is a major energy consumer during the system operation [31]. Simulations shall be able to help the designer find a balance between the wireless communication and the local processing. Therefore, WSN simulations require designers to integrate the node system and the network simulation.

A common way to evaluate the WSN system is to add sensor node models to the network simulators (e.g., NS-2 [15] and OMNeT++ [32]). There are two kinds of node model: node models implemented by the network simulators and node emulators. The latter refers to the instruction-level simulators of the nodes' microcontrollers or operating system emulators.

Besides adding node models to the network simulators, we can also model the network in the node system design tools (e.g., SCNSL [33]) or in the node emulator (e.g., Avrora [34]). Therefore, as illustrated in Figure 2, the existing efforts in WSN evaluation can be divided into four categories: network simulators with node models (NSNM), network simulators with node emulators (NSNE), node system simulator with network models (NSSNM), and node emulators with network models (NENM).

Network simulators with node models emphasize more on discrete event scheduling, the radio medium, network modeling, and perhaps the sleep duty cycles of the sensor node. Network modeling is the predominate object. Many node models implemented by this kind of simulator are simple power and estimated timing profiles.

Network simulators with node emulators integrate the advantages of both the network simulators and node emulators. The network simulator provides the detailed network model. The node emulator gives accurate timing information of the software execution because they simulate the system performance with instruction cycle granularity. However, the interconnection between the network simulator and node emulator may take much time.

In node system simulators with network models, the node system is often modeled by system-level description languages (SLDL), such as SystemC [35]. They have a simulation kernel which supports modeling the concurrency and synchronization among different hardware components. SLDL can also model the software, which allows the HW/SW co-design and co-simulation. It models the node hardware in different abstraction levels with different degrees of detail (e.g., system level, transaction level, and register transfer level). Compared with network simulators with node models, node system simulators with network models can provide more accurate power and timing modeling.

The node emulators with network models can be divided into two different sets: ISSs for special microcontrollers or processors, and emulators designed to emulate the execution of the application code of an operating system (e.g., TinyOS [36], SOS [37], and Contiki [38]). They can provide high timing accuracy of software execution. The embedded software developed for physical platforms can be executed directly in the simulation framework with little or no modifications.

Each type of simulation tools in these four categories possesses its unique feature and emphasizes on some special requirements of simulation and modeling for wireless networked embedded systems. The unique features of each type are summarized in Table 1.

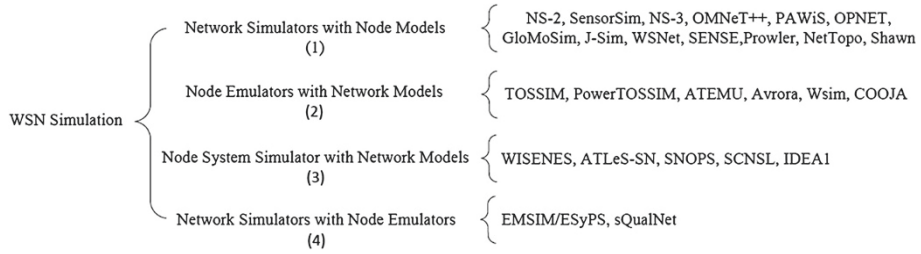

Figure 2 Taxonomy of simulation tools for wireless sensor networks.

Table 1 Unique features of simulation tools in different categories

Simulators	Advantages	Disadvantages
Network simulators with node models	Network modeling Radio medium Scalability	Simple power profile Simple timing profiles
Network simulators with node emulators	Network modeling Detailed power profile Detailed timing profiles	Scalability
Node system simulator with network models	Network modeling Radio medium Scalability	Moderate timing accuracy Moderate power accuracy
Node emulators with network models	High timing accuracy High energy accuracy	Simple network modeling Simple radio medium Scalability

Survey on WSN simulation tools

In this section, we will analyze the existing simulation tools according to classification scheme of the taxonomy presented in the 'Taxonomy on WSN simulation tools' section. The existing simulation tools are divided to four groups. Many simulators will be studied to demonstrate the common features of each category.

Network simulators with node models

Many general-purpose network simulators, such as NS-2 [15], OMNeT++ [32], OPNET [39], GloMoSim [40], and J-Sim [13], have been utilized in WSN simulations. Some extensions have been applied to them to introduce the WSN specific characteristics. Besides the extensions to general-purpose network simulators, some WSN-specific network simulators have also been developed.

NS-2 [15] is a discrete event, object-oriented, general-purpose network simulator. Simulations are written by C++ and OTcl (Object-oriented Tcl). In general, C++ is used for implementing protocols and extending the library. OTcl is used to create and control the simulation environment. Its extensibility has been a major contributor to its success, with protocol implementations being widely produced and developed by the research community. The performance of wireless LAN is studied in [41,42]. According to [43], it is the most used simulator in mobile ad hoc network (MANET) research. Regarding WSN, it includes many *ad hoc* and WSN-specific protocols [8]. An IEEE 802.15.4 model is developed in [44]. An energy-efficient routing in WSNs [45] and directional routing in vehicular networks [46] are evaluated using NS-2 simulations based on a simple energy model. However, NS-2 does not scale well in terms of memory usage and simulation time [47]. It also lacks detailed support to measure the energy utilization of different hardware, software, and firmware components of a WSN node [48]. SensorSim [12] is built on top of an NS-2 802.11 network model. It models the sensor node in two parts: software model (function model) and hardware model. The power models of different hardware components have been implemented. The state of the hardware model is changed based on the function that is carried out by the software model. Therefore, the power consumption of the whole network can be simulated. In addition, SensorSim can be interacted with real nodes. However, the CPU and sensor models have not been implemented. Furthermore, IEEE 802.11 is designed for high-speed connectivity and not optimized for WSNs.

OMNeT++ [32] is a component-based network simulator, with an Eclipse-based integrated development environment (IDE) and a graphical runtime environment. The IDE supports all stages of a simulation project: developing, building, configuring, and running simulation models and analyzing results. OMNeT++ consists of modules that communicate with message passing. Simple modules implement the atomic behavior of a model, e.g., a particular protocol. Multiple simple modules can be linked together and form a compound module. OMNeT++ provides the infrastructure to assemble simulations from these modules and configure them (NED language). OMNeT++ can be extended easily by interfaces for real-time simulation, emulation, parallel distributed simulation, SystemC integration, and so on. As OMNeT++ is becoming more popular, many contributions have been added to it. The Mobility Framework (MF) [49] supports simulations of wireless and mobile networks within OMNeT++. MF includes an 802.11 model. It can be seen as the first start point of WSN modeling by OMNeT++. An IEEE 802.15.4 implementation by OMNeT++ can be

found in [50]. PAWiS [48,51] is an OMNeT++ based WSN simulator. Its architecture is similar to SensorSim. It can evaluate the power consumption of WSN systems with many levels of accuracy which can still be balanced with complexity. The model programmer has to insert special framework requests to the CPU module to simulate the execution time and power consumption. These requests include the estimated execution time of the firmware code on the CPU.

WSNet [52] is a modular-based, event-driven, high-level wireless network simulator. It is composed of many blocks that model the properties of sensor nodes and radio medium. The sensor node model includes the hardware and software abstraction and node behavior modeling (e.g., mobility). WSNet can be used to evaluate the high-level design, such as traffic pattern, application dimensioning, and protocol parameter tuning. It is one of the two simulators in Worldsens [52], an integrated environment for development and rapid prototyping of wireless sensor network applications. Worldsens also includes a low-level simulator to enable the refinement of WSN application development. The cycle accurate simulator, WSim, will be introduced in the 'Node emulators with network models' section.

SENSE [53] is another component-based simulator developed by C++. It models various network devices as a collection of components. Connections between each component are in the format of input and output ports. Packets are created, transmitted, and received by components through the ports. Through its component-based model, SENSE can be extended easily. A new component can replace an old one if they have compatible interfaces; inheritance is not required. SENSE also supports the parallel simulation, which is provided as an option to the users.

GloMoSim [40] is a parallel simulator for WSNs. GloMoSim allows the users to select sequential or one of the three available parallel synchronization algorithms (null message protocol, conditional event protocol, and accelerated null message protocol). Once a parallel algorithm is selected, the analyst can additionally indicate the mapping strategy and number of processors. Taking advantage of parallel simulation, GloMoSim has been shown to scale to 10,000 nodes [54]. QualNet [55] is a commercial derivative of GloMoSim. It has extended GloMoSim to other networks, such as satellite, cellular, and sensor networks. ZigBee protocol model is also provided.

Prowler [56] is an event-driven network simulator running in Matlab environment. Benefits gained from Matlab environment are easy prototyping of applications and GUI interface. Prowler is capable of simulating the radio transmission, propagation, and the MAC-layer operation in *ad hoc* networks. The radio models are based on specific signal strength calculations combined with random errors. Prowler is well suited for protocol and algorithm development. However, it does not have sensor node energy modeling.

NetTopo [57] is an integrated WSN-specific network simulator that provides the simulation of virtual WSN and the visualization of real testbeds. It also supports the interaction between the simulated WSN and real testbeds. Shawn [17] is another open-source discrete event simulator written in C++. It is suitable for large-scale network simulations benefitting from the high-level abstraction.

Some customized simulators are developed to study some particular problems in WSNs. For example, the sensor placement problem [58] and routing protocol design [59] are studied by Matlab. Sometimes, the collected traces from the real deployments [60,61] are imported into the simulation models to reproduce the transmission behaviors of links and paths in WSNs. A discrete event simulator is developed in Java to study the MAC performance of WSNs [62]. The data aggregation and coverage problems for WSNs are studied by C++ based simulations in [63,64]. The channel assignment problem is studied in [65]. We classify these simulators as network simulators.

The main advantages of network simulator are that they usually have a rich library of the radio modules and protocol implementations. Many contributions to these tools are carried out ceaselessly. For example, the performances of NS-2 in the aspects of scalability and extensibility are improved by its successor, NS-3 [66]. It also simplified the model implementation by choosing C++ as the sole development language, and the usage of Python scripting language can be optionally enable [67]. However, the network simulators are dedicated to model the network. It may be not the best way to model the node system since they are normally incapable to model the concurrency within the node and provide a direct path to HW/SW synthesis [33]. The energy consumption is usually based on some assumptions or estimations of the software execution, for example, the processor and RF transceiver of a sensor node have the same operating state, but in fact the processor can be in sleep mode when the RF transceiver is listening to the channel and woken up by the RF transceiver when the latter receives a packet. Most of the energy models for simulations at network level are not complete. The problems are the following:

- Some energy models assume that the radio consumes the same power in idle listening as in receiving state and they are ignoring the energy consumption in sleeping state.
- Few simulation models take into account the transition energy cost for switching between the radio operational states.

- The energy consumption of the microcontroller is frequently not considered, or the power profile is very simple (just active and inactive states).

Node emulators with network models

Two kinds of node emulator, operating system emulator and instruction set simulator, are studied separately in this section. One special simulator providing both features will also be presented.

TOSSIM [16] and PowerTOSSIM [25] are two emulators designed to emulate the execution of TinyOS [36]. Software development for WSN can be simplified by using these emulators. They permit developing algorithms, studying system behaviors, and observing interactions among the nodes in a controlled environment. The application code of TinyOS can be compiled to the simulation framework by only replacing a few low-level TinyOS components that deal with hardware. TOSSIM can capture the behavior of the network of thousands of TinyOS nodes at bit granularity. TOSSIM allows the developer to easily transition between running an application on motes and in the simulation environment. PowerTOSSIM is an extension to TOSSIM in evaluation of the power consumption. The main problem of such frameworks is that the user is constrained to a specific platform (typically MICA motes) and a single programming language (typically TinyOS/NesC) [51]. In addition, TOSSIM loses the fine-grained timing and interrupts properties of the code that can be important when the application runs on the hardware and interacts with other nodes [34].

ATEMU [68] is an instruction-level, cycle-accurate emulator for WSN written in C. It simulates programs of each individual node with accuracy down to the clock cycle. Its core is an ISS. Along with support for the AVR processor, it also includes support for other peripheral devices on the MICA2 sensor node platform, such as the transceiver. ATEMU provides a GUI, called Xatdb, which provides users a complete system for debugging and monitoring the execution of their code. Avrora [34], written in Java, improves the performance of ATEMU in the scalability aspect. Avrora can scale to networks of up to 10,000 nodes. Both ATEMU and Avrora provide a high behavioral and timing accuracy of the WSN programs. Moreover, they are both language and operating system independent. The main disadvantage of such frameworks is that they only support systems based on components that have already existed, e.g., memories and processors, like MICA motes. Unfortunately, they do not cover systems containing new hardware blocks.

WSim is the low-level simulator in Worldsens [52]. It is based on cycle-accurate full platform simulation using microprocessor instruction driven timings. It provides many hardware block descriptions of components on the chip level. A sensor node platform can be built by describing the physical interconnection among these components. WSim can also handle real target binary code simulation and debugging. The time resolution can be at nanosecond level. By combining WSim and WSNet, Worldsens can provide a complete design flow of WSN application, from the high-level design choices down to the target code implementation, debug, and performance analysis.

Fummi et al. [69] have developed an energy-aware simulator by integrating an ISS of the node's microcontroller and a functional SystemC model of the network module on SCNSL [33]. SCNSL is a networked embedded system simulator, which will be introduced in the 'Node system simulator with network models' section. μCsim is used as the ISS for the Intel 8051 microcontroller of the Texas Instruments CC2430F128 chip. Using ISS makes it possible to run the exact binary embedded software on the simulated hardware platform. The SystemC kernel is modified to communicate with the ISS through inter-process communication primitives (e.g., a socket or shared memory).

COOJA [70] is a Java-based simulator that provides both the operating system emulation and the instruction set emulation in a single framework. The Contiki operating system [38] can be compiled to the simulation framework. It executes native code by making Java Native Interface (JNI) calls from the Java environment to a compiled Contiki system. MSPSim [71] is used as the instruction set simulator in the COOJA. MSPSim is also written in Java. It supports the Texas Instruments MSP430 microcontroller and includes some hardware peripherals such as sensor, communication ports, LEDs, and sound devices. Recently, the COOJAMSPSim platform [72] has been extended to support the TinyOS. The interoperability testing of nodes with different operating systems is realized.

The main advantage of using such tools is that the code used for emulation can also run on the real node, which reduces the effort to rewrite the code. In addition, they often provide detailed information about resource utilization. The main problems are that they are always constrained to specific hardware platforms or operating systems. Because much detail of cycle-accurate level is considered, they cannot scale as well as the node system models at system level. Moreover, for new applications, it may take more time to develop the final executable code than to abstract the applications at the beginning of system design.

Node system simulator with network models

Wireless Sensor Network Simulator (WISENES) [73] is developed in specification and description language (SDL) [74], which is a high-level abstraction language widely

used in communication protocol design and can be converted to C code automatically. The key feature of WISENES is that its simulation models are reusable in the embedded software design for the final system. However, WISENES only contributes to the software implementation. SDL is unsuitable to model synchronous digital circuits because the SDL system behavior is defined as a network of extended finite state machines that communicate with each other using asynchronous signals [75]. On the other hand, SystemC provides native supports of HW/SW co-simulation.

Virk et al. [76] have developed a SystemC-based modeling framework for WSN. It models the applications, real-time operating systems, sensors, processor, and transceiver at node level and signal propagations at network level. It is the first work using SystemC in WSN simulation, but the simulation result is simple. Only a MAC behavior (states of the sending and receiving tasks) waveform has been presented in [76].

ATLeS-SN (Arizona Transaction-Level Simulator for Sensor Network) [77] is a transaction-level modeling (TLM)-based sensor network simulation environment developed in SystemC. It models a sensor node in three components: application specification, network stack implementation, and sensor system. The physical channel is modeled as a component. It provides an interface that can be called from sensor nodes. ATLeS-SN demonstrated the feasibility of using TLM for sensor network application, but no standard networking protocol has been implemented.

The SNOPS framework [78] is another TLM-based WSN simulator. A sensor node transmits or receives a data packet to or from an environment model by transaction exchanges. The SNOPS framework [78] requires 49.7% less simulation time than PAWiS [48].

SystemC Network Simulation Library (SCNSL) [33] is a networked embedded system simulator, written in SystemC and C++. It includes three modules: node (SystemC), node-proxy (SystemC), and network (C++). During the initialization of simulation, each node registers its information (e.g., location, TX power, and RX sensitivity) at a network class which maintains the network topology and transmits packets to other nodes. The node-proxy is an interface between the network and nodes. By

Table 2 Features of classical simulators for wireless sensor networks

Simulators	Language	WSN specifications	Software modeling	Hardware modeling	Energy aware	Heterogeneity support	Easy to use
NS-2 [15]	C++ OTcl	IEEE 802.15.4 [44] SensorSim [12]	No	No	Low	Perfect	Good
OMNet++ [32]	C++	IEEE 802.15.4 [50] PAWiS [48]	No	No	Low	Perfect	Good
WSNet [52]	C	IEEE 802.15.4	No	No	Low	Good	General
SENSE [53]	C++	IEEE 802.11	No	No	Low	Perfect	General
GloMoSim [40]	PARSEC	NaN	No	No	Low	Good	General
Prowler [56]	Matlab	Yes	No	No	Low	Good	General
TOSSIM [16]	C	TinyOS	Yes	Power-TOSSIM [25]	Perfect	General	Good
ATEMU [68]	C	Yes	ISS	No	Perfect	Low	General
WSim [52]	C	Yes	ISS	No	Perfect	Low	General
COOJA [70]	Java	Contiki	Yes	No	Perfect	General	General
WISENES [73]	SDL[74]	Yes	Yes	No	Good	Good	General
ATLeS-SN [77]	SystemC	Yes	Yes	No	Good	Good	General
SCNSL [33]	SystemC	No	No	No	Low	Good	General
IDEA1 [80]	Yes	IEEE 802.15.4	Yes	Yes	Good	Good	Good
sQualNet [86]	PARSEC	QualNet [55]	Yes	No	Good	Low	General

using node-proxy, nodes can be designed as pure SystemC modules so as to exploit all advantages of SystemC in HW/SW co-design and verification. SCNSL demonstrates a great perspective for system-level simulation of WSN system, but it still has some limitations such as node-level simulation without any specific hardware platform or energy model.

IDEA1 [1,79,80] is based on the SCNSL library of alpha version. The network model of IDEA1 is inherited from SCNSL; however, many contributions have been developed. Emphasizing the modular design, IDEA1 models a sensor node exactly according to its hardware architecture. Each hardware component is modeled as an individual module. By doing this, the behaviors of hardware components can be accurately captured, which is the basis of energy consumption estimation. An energy model [81] has been developed to enable the accurate energy consumption prediction. It has been calibrated by some experimental measurements. Both free space and International Telecommunication Union (ITU) indoor propagation models are implemented. The simulation results of IDEA1 have been validated by a testbed consisting of nine nodes and compared with NS-2. Based on the efficient simulation kernel of SystemC and optimized model implementation, the simulation speed of IDEA1 is twice faster than NS-2. Using IDEA1, the performance of IEEE 802.15.4 MAC protocols is studied in [82] and a real application of WSNs on vibration measurements is studied in [83].

These simulators scale better than node emulators since they usually model sensor node at system level. Additionally, new hardware and software modules can be easily added to the existing library. However, they are normally based on a special HW/SW description language which poses an extra learning burden to the new users.

Network simulators with node emulators

Two main simulators have been developed in this category. Park et al. [84] have developed a unified network and node level simulation framework. They developed the Embedded Systems Power Simulator (ESyPS) by integrating sensor and radio modules into EMSIM [85]. EMSIM is an energy simulation framework for embedded systems featured in StrongARM microprocessor and Linux OS. Then, they integrated the ESyPS with SensorSim [12]. The framework can explore the interactions between network level and node level.

Another example is sQualNet [86], which is a scalable and extensible sensor network simulation framework built on top of QualNet [55]. It uses QualNet as the network simulator and provides the emulation of the SOS operating system [37]. sQualNet allows using the QualNet's detailed models of channel, propagation, mobility, etc. The user also can use the rich protocol suite for other kinds of networks to model heterogeneous sensor networks. sQualNet introduces a sensor stack parallel to the networking stack and provides accurate simulation models for various layers in the sensor and networking stack.

These two simulators integrate the advantages of both the network simulators and node emulators. They provide accurate results about the energy consumption of the whole network. However, they are both constrained to a particular hardware and operating system. Moreover, interactions between the network simulator and the node emulator have to be well maintained, which increases the simulation time and impacts the scalability.

Summary

In the above sections, we use the classical simulators in each category to analyze their unique features. We found that due to its specific target and design, each simulator possesses its own characteristic. In summary, the main features of some typical simulators are summarized in Table 2. The grade of the performance in terms of *scalability*, *energy aware*, *extensibility*, *heterogeneity support*, and *easy to use* is presented by this granularity: perfect, good, general, and low. NaN means that we have not found the necessary information.

Conclusion

In this paper, we study the modeling and simulation of WSN systems. Based on a typical WSN system model, a taxonomy of WSN simulations is proposed. According to the taxonomy, a survey of the existing simulation tools for WSN is made. Most of the significant existing simulation tools with relatively widespread uses have been studied. We believe that the survey is comprehensive enough to prove that almost all the simulation tools for WSN can be divided into one of the four categories in our classification scheme. Some simulators may be out of maintenance at this moment; however, it is worth to analyze them here so as to illustrate the evolution process of WSN simulation techniques. This paper can be used by WSN designers to better understand the state-of-art of WSN simulation and to find an appropriate simulator or develop their own custom simulator for their special applications.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Lyon Institute of Nanotechnology, University of Lyon, Lyon 69007, France. [2]Present address: School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore.

Received: 10 February 2014 Accepted: 4 June 2014
Published: 3 July 2014

References

1. W Du, F Mieyeville, D Navarro, I O'Connor, IDEA1, A validated SystemC-based system-level design and simulation environment for wireless sensor networks. EURASIP J. Wireless Commun. Netwo, 1–20 (2011)
2. Crossbow Technology, Inc., MICAz datasheet [Online]. Available: http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf. Accessed 20 September 2009
3. Crossbow Technology, Inc., TelosB datasheet [Online]. Available: www.willow.co.uk/TelosB_Datasheet.pdf. Accessed 20 September 2009
4. Crossbow Technology, Inc., Imote2 Hardware Reference Manual [Online]. Available: http://wsn.cse.wustl.edu/images/e/e3/Imote2_Datasheet.pdf. Accessed 20 September 2009
5. H Dubois-Ferrire, R Meier, L Fabre, P Metrailler, TinyNode: a comprehensive platform for wsn applications, in *ACM/IEEE IPSN* (Nashville, USA, 19–21 April 2006)
6. Ed. IEEE IEEE Computer Society, *IEEE Standard for Information Technology-Telecommunications and information exchange between systems- Local and metropolitan area networks- Specific requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs),* Institute of Electrical and Electronics Engineers, 2006
7. ZigBee Alliance, ZigBee specification - Document 053474r17 (2007). [Online]. Available: http://www.zigbee.org/. Accessed 20 September 2009
8. E Egea-López, J Vales-Alonso, AS Martínez-Sala, P Pavón-Marõio, J García-Haro, Simulation tools for wireless sensor networks, in *Proc. Int. Symp. Perform Eval. Comput. Telecommun. Syst* (Philadelphia, 24–28 July 2005)
9. W Du, D Navarro, F Mieyeville, A simulation study of IEEE 802.15.4, sensor networks in industrial applications by system-level modeling, in *Proc. of the Fourth International Conference on Sensor Technologies and Applications (SENSORCOMM)* (IEEE, 2010 Venice/Mestre, Italy, 18–25 July 2010), pp. 311–316
10. J Haase, JM Molina, D Dietrich, Power-aware system design of wireless sensor networks: power estimation and power profiling strategies. IEEE Trans. Ind. Inform. **7**(4), 601–613 (2011)
11. I Stojmenovic, Simulations in wireless sensor and ad hoc networks: matching and advancing models, metrics, and solutions. IEEE Commun Mag. **46**(12), 102–107 (2008)
12. S Park, A Savvides, MB Srivastava, Sensorsim: a simulation framework for sensor networks, in *Proc. of the 3rd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems,* ser MSWIM '00 (ACM New York, 2000), pp. 104–111
13. A Sobeih, W-P Chen, JC Hou, L-C Kung, N Li, H Lim, H-Y Tyan, H Zhang, J-Sim: a simulation environment for wireless sensor networks, in *Proc. of the 38th Annual Symposium on Simulation,* ser. ANSS '05 (IEEE Computer Society Washington, DC, 2005), pp. 175–187
14. H Sundani, H Li, V Devabhaktuni, M Alam, P Bhattacharya, Wireless sensor network simulators a survey and comparisons. Int J. Comput. Netw. **2**, 249–265 (2010)
15. K Fall, K Varadhan, The ns Manual (formerly ns Notes and Documentation), (Jan. 2009). [Online]. Available: http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf. Accessed 20 September 2009
16. P Levis, N Lee, M Welsh, D Culler, TOSSIM: accurate and scalable simulation of entire tinyos applications, in *Proc. of the 1st Int Conf. on Embedded Networked Sensor Systems,* ser. SenSys '03 (ACM New York, 2003), pp. 126–137
17. SP Fekete, A Kroller, S Fischer, D Pfisterer, Shawn: the fast, highly customizable sensor network simulator, in *Fourth International Conference on Networked Sensing Systems, 2007. INSS'07* (IEEE, 2007 Braunschweig, Germany, 6–8 June 2007), pp. 299–299
18. M Li, Z Li, AV Vasilakos, A survey on topology control in wireless sensor networks: taxonomy, comparative study, and open issues, in *Proc. IEEE*, (2013)
19. AV. Vasilakos, Y Zhang, T Spyropoulos, *Delay Tolerant Networks Protocols and Applications*. (CRC Press, Boca Raton, 2012)
20. T Spyropoulos, RN Rais, T Turletti, K Obraczka, AV Vasilakos, Routing for disruption tolerant networks: taxonomy and design. Wireless Netw. **16**(8), 2349–2370 (2010)
21. M Youssef, M Ibrahim, M Abdelatif, L Chen, AV Vasilakos, Routing metrics of cognitive radio networks: a survey. IEEE Commun Surv. Tutor. **16**(1), 92–109 (2013)
22. X Wang, AV Vasilakos, M Chen, Y Liu, TT Kwon, A survey of green mobile networks: opportunities and challenges. Mobile Netw Appl. **17**(1), 4–20 (2012)
23. W Du, D Navarro, F Mieyeville, F Gaffiot, Towards a taxonomy of simulation tools for wireless sensor networks, in *Proc. of the 3rd Int. ICST Conference on Simulation Tools and Techniques* (ser. SIMUTools '10, 2010 Malaga, Spain 15–19 March 2010), pp. 52:1–52:7
24. M Varshney, R Bagrodia, Detailed models for sensor network simulations and their impact on network performance, in *Proc. of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems,* ser. MSWiM '04 (ACM New York, 2004), pp. 70–77
25. V Shnayder, M Hempstead, B-r Chen, GW Allen, M Welsh, Simulating the power consumption of large-scale sensor network applications, in *Proc. of the 2nd Int. Conf. on Embedded Networked Sensor Systems,* ser SenSys '04 (ACM New York, 2004), pp. 188–200
26. L Girod, T Stathopoulos, N Ramanathan, J Elson, D Estrin, E Osterweil, T Schoellhammer, A system for simulation, emulation, and deployment of heterogeneous sensor networks, in *Proc. of the 2nd Int. Conf. on Embedded Networked Sensor Systems,* ser. SenSys '04 (ACM New York, 2004), pp. 201–213
27. GV Merrett, NM White, NR Harris, BM Al-Hashimi, Energy-aware simulation for wireless sensor networks, in *Proc. of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks,* ser. SECON'09 (IEEE Press Piscataway, 2009), pp. 64–71
28. F Fummi, M Loghi, G Perbellini, M Poncino, SystemC co-simulation for core-based embedded systems. Des. Automation Embedded Syst. **11**, 141–166 (2007). doi:10.1007/s10617-007-9006-7
29. L Séméria, A Ghosh, Methodology for hardware/software co-verification in C/C++ (short paper), in *Proc. of the 2000 Asia and South Pacific Design Automation Conference,* ser. ASP-DAC '00 (ACM New York, 2000), pp. 405–408
30. M Degermark, A Brodnik, S Carlsson, S Pink, Small forwarding tables for fast routing lookups, in *Proc. of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication,* ser. SIGCOMM '97 (ACM New York, 1997), pp. 3–14
31. V Raghunathan, C Schurgers, S Park, MB Srivastava, Energy-aware wireless microsensor networks. Signal Process. Mag. IEEE. **19**(2), 40–50 (2002)
32. A Varga, The OMNet++ discrete event simulation system, in *Proc. Eur. Simul. Multiconference (ESM'2001)* (Prague, Czech Republic, 6–9 June 2001)
33. F Fummi, D Quaglia, F Stefanni, A SystemC-based framework for modeling and simulation of networked embedded systems, in *Proc. of the Forum on Specification and Design Languages* (Stuttgart, Germany, 23–25 September 2008), pp. 49–54
34. BL Titzer, DK Lee, J Palsberg, Avrora: scalable sensor network simulation with precise timing, in *Proc. of the 4th International Symposium on Information Processing in Sensor Networks,* ser. IPSN '05 (IEEE Press Piscataway, 2005)
35. Ed. IEEE IEEE-SA Standards Board, *IEEE Std 1666 - 2005 IEEE Standard SystemC Language Reference Manual*. Institute of Electrical and Electronics Engineers, 2006
36. J Hill, R Szewczyk, A Woo, S Hollar, D Culler, K Pister, System architecture directions for networked sensors. ACM SIGPLAN Notices. **35**, 93–104 (2000)
37. C-C Han, R Kumar, R Shea, E Kohler, M Srivastava, A dynamic operating system for sensor nodes, in *Proc. of the 3rd Int. Conf. on Mobile Systems, Applications, and Services,* ser. MobiSys '05 (ACM New York, 2005), pp. 163–176
38. A Dunkels, B Gronvall, T Voigt, Contiki - a lightweight and flexible operating system for tiny networked sensors, in *Proc. of the 29th Annual IEEE Int. Conf. on Local Computer Networks,* ser. LCN '04 (IEEE Computer Society Washington, DC, 2004), pp. 455–462
39. OPNET Technologies, Inc., OPNET (2006). [Online]. Available: http://www.opnet.com/. Accessed 20 September 2009
40. X Zeng, R Bagrodia, M Gerla, GloMoSim: a library for parallel simulation of large-scale wireless networks, in *Proc. of the Twelfth Workshop on Parallel and Distributed Simulation,* ser. PADS '98 (IEEE Computer Society Washington, DC, 1998), pp. 154–161
41. W Du, M Li, Harnessing mobile multiple access efficiency with location input, in *Proc. of the IEEE 33rd International Conference on Distributed*

Computing Systems (ICDCS) (IEEE, 2013 Philadelphia, USA, 8–11 Jul 2013), pp. 246–255
42. W Du, M Li, J Lei, CO-MAP: improving mobile multiple access efficiency with location input. IEEE Trans. Wireless Commun (2014)
43. S Kurkowski, T Camp, M Colagrosso, Manet simulation studies: the incredibles. SIGMOBILE Mob. Comput. Commun. Rev. **9**, 50–61 (2005)
44. J Zheng, MJ Lee, Will IEEE 802.15.4 make ubiquitous networking a reality?: a discussion on a potential low power, low bit rate standard. Commun. Mag. IEEE. **42**(6), 140–146 (2004)
45. N Chilamkurti, S Zeadally, AV Vasilakos, V Sharma, Cross-layer support for energy efficient routing in wireless sensor networks. J. Sensors. **2009**, 9 (2009)
46. Y Zeng, K Xiang, D Li, AV Vasilakos, Directional routing and scheduling for green vehicular delay tolerant networks. Wireless Netw. **19**(2), 161–173 (2013)
47. V Naoumov, T Gross, Simulation of large ad hoc networks, in *Proc. of the 6th ACM International Workshop on Modeling analysis and Simulation of Wireless and Mobile Systems,* ser. MSWIM '03 (ACM New York, 2003), pp. 50–57
48. J Glaser, D Weber, SA Madani, S Mahlknecht, Power aware simulation framework for wireless sensor networks and nodes. EURASIP J. Embedded Syst. **2008**, 3:1–3:16 (2008)
49. W Drytkiewicz, S Sroka, V Handziski, A Koepke, H Karl, A mobility framework for OMNeT++, in *Proc. of the 3rd Int. OMNeT++ Workshop* (Budapest, Hungary, 28 January 2003)
50. F Chen, N Wang, R German, F Dressler, Simulation study of IEEE 802.15.4 LR-WPAN for industrial applications. Wireless Commun Mobile Comput. **10**(5), 609–621 (2010)
51. D Weber, J Glaser, S Mahlknecht, Discrete event simulation framework for power aware wireless sensor networks, in *Proc. of the 5th Int Conf. on Industrial Informatics* (Vienna, Austria, 23–26 July 2007), pp. 335–340
52. G Chelius, A Fraboulet, E Fleury, Worldsens: development and prototyping tools for application specific wireless sensors networks, in *Proc. of the International Conference on Information Processing in Sensor Networks* (Boston, USA, 2007), pp. 176–185
53. G Chen, J Branch, M Pflug, L Zhu, B Szymanski, ed. by Szymanski BK, Yener B, Sense: a wireless sensor network simulator, in *Advances in Pervasive Computing and Networking* (Springer USA, 2005), pp. 249–267. doi:10.1007/0-387-23466-7_13
54. M Takai, R Bagrodia, K Tang, M Gerla, Efficient wireless network simulations with detailed propagation models. Wirel. Netw. **7**, 297–305 (2001)
55. Scalable Network Technologies. QualNet. [Online]. Available: http://www.scalable-networks.com. Accessed 20 September 2009
56. G Simon, P Volgyesi, M Maroti, A Ledeczi, Simulation-based optimization of communication protocols for large-scale wireless sensor networks, in *2003 IEEE Aerospace Conference,* Big Sky, MT, March 2003
57. L Shu, C Wu, Y Zhang, J Chen, L Wang, M Hauswirth, NetTopo: beyond simulator and visualizer for wireless sensor networks. SIGBED Rev. **5**, 2:1–2:8 (2008)
58. W Du, Z Xing, M Li, B He, LHC Chua, H Miao, Optimal sensor placement and measurement of wind for water quality studies in urban reservoirs, in *Proc. of the ACM/IEEE 13th International Symposium on Information Processing in Sensor Networks (IPSN)* (IEEE Press, 2014 Berlin, Gemany, 15–17 April, pp. 167–178
59. Y Liu, N Xiong, Y Zhao, AV Vasilakos, J Gao, Y Jia, Multi-layer clustering routing algorithm for wireless vehicular sensor networks. IET Commun. **4**(7), 810–816 (2010)
60. Z Li, M Li, Y Liu, Towards energy-fairness in asynchronous duty-cycling sensor networks. ACM Trans. Sensor Netw. **10**(3), 38:1–38:26 (2014). [Online]. Available: http://doi.acm.org/10.1145/2490256
61. Z Li, Y Liu, M Li, J Wang, Z Cao, Exploiting ubiquitous data collection for mobile users in wireless sensor networks. IEEE Trans. Parallel Distributed Syst. **24**(2), 312–326 (2013)
62. Y Xiao, M Peng, J Gibson, GG Xie, D-Z Du, AV Vasilakos, Tight performance bounds of multihop fair access for mac protocols in wireless sensor networks and underwater sensor networks. IEEE Trans Mobile Comput. **11**(10), 1538–1554 (2012)
63. L Xiang, J Luo, AV Vasilakos, Compressed data aggregation for energy efficient wireless sensor networks, in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON) 2011 8th Annual IEEE Communications Society Conference on* IEEE, 2011 Salt Lake City, USA, 27–30 June 2011), pp. 46–54
64. M Peng, H Chen, Y Xiao, S Ozdemir, AV Vasilakos, J Wu, Impacts of sensor node distributions on coverage in sensor networks. J Parallel Distributed Comput. **71**(12), 1578–1591 (2011)
65. H Cheng, N Xiong, AV Vasilakos, L Tianruo Yang, G Chen, X Zhuang, Nodes organization for channel assignment with topology preservation in multi-radio wireless mesh networks. Ad Hoc Netw. **10**(5), 760–773 (2012)
66. Weingärtner E, H Vom Lehn, K Wehrle, Proc. of the 2009 IEEE Int. Conf. on Communications, *ser. ICC'09* (IEEE Press Piscataway, 2009), pp. 1287–1291
67. JL Font, P Iñigo, M Domínguez, JL Sevillano, C Amaya, Analysis of source code metrics from ns-2 and ns-3 network simulators. Simul. Model. Prac. Theory. **19**(5), 1330–1346 (2011)
68. J Polley, D Blazakis, J McGee, D Rusk, JS Baras, ATEMU: a fine-grained sensor network simulator, in *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004* (Santa Clara, USA, 4–7 October 2004), pp. 145–152
69. F Fummi, G Perbellini, D Quaglia, A Acquaviva, Flexible energy-aware simulation of heterogeneous wireless sensor networks, in *Proc. of the Conference on Design, Automation and Test in Europe,* ser. DATE '09. 3001, Leuven, Belgium (European Design and Automation Association, Belgium, 2009), pp. 1638–1643
70. F Österlind, A Dunkels, J Eriksson, N Finne, T Voigt, Cross-level sensor network simulation with COOJA, in *Proc. of the First IEEE Int. Workshop on Practical Issues in Building Sensor Network Applications (SenseApp '06)* (Tampa, FL, Nov, 2006)
71. J Eriksson, A Dunkels, N Finne, F Österlind, T Voigt, N Tsiftes, Demo abstract: Mspsim - an extensible simulator for msp430-equipped sensor boards, in *Proc. of the 5th European Conference on Wireless Sensor Networks* (Zurich, Switzerland, 13–15 February 2006)
72. J Eriksson, F Österlind, N Finne, N Tsiftes, A Dunkels, T Voigt, R Sauter, PJ Marrón, COOJA/MSPSim: interoperability testing for wireless sensor networks, in *Proc. of the 2nd Int. Conf. on Simulation Tools and Techniques,* ser. Simutools '09. ICST, Brussels, Belgium (ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Belgium, 2009), pp. 27:1–27:7
73. M Kuorilehto, M Hännikäinen, TD Hämäläinen, Rapid design and evaluation framework for wireless sensor networks. Ad Hoc Netw. **6**, 909–935 (2008)
74. SDL Forum Society homepage. [Online]. Available: http://www.sdl-forum.org/. Accessed 20 September 2009
75. M Haroud, A Biere, Hw accelerated ultra wide band MAC protocol using SDL and SystemC, in *Proc. of the 10th Int. Conf. on Architectural Support for Programming Languages and Operating Systems,* ser. ASPLOS-X (ACM New York, 2002), pp. 85–95
76. K Virk, K Hansen, J Madsen, System-level modeling of wireless integrated sensor networks, in *Proc. of the Int. Symposium on System-on-Chip* (Tampere, Finland, 15–17 November 2005), pp. 179–182
77. J Hiner, A Shenoy, R Lysecky, S Lysecky, AG Ross, Transaction-level modeling for sensor networks using SystemC, in *Proc. of the 2010 IEEE Int. Conf. on Sensor Networks, Ubiquitous, and Trustworthy Computing,* ser. SUTC '10 (IEEE Computer Society Washington, DC, 2010), pp. 197–204
78. M Damm, J Moreno, J Haase, C Grimm, Using transaction level modeling techniques for wireless sensor network simulation, in *Proc. of the Conference on Design, Automation and Test in Europe,* ser. DATE '10 (Dredsen, Germany, 8–12 March 2010), pp. 1047–1052
79. W Du, F Mieyeville, D Navarro, IDEA1: a SystemC-based system-level simulator for wireless sensor networks, in *Proc. of the IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS)* (Beijing, China, 25–27 June 2010), pp. 618–622
80. W Du, D Navarro, F Mieyeville, I O'connor, IDEA1: a validated SystemC-based simulator for wireless sensor networks, in *Proc. of IEEE 8th International Conference on Mobile Adhoc and Sensor Systems (MASS)* (Wuhan, China, 12–14 August 2011), pp. 825–830
81. W Du, F Mieyeville, D Navarro, Modeling energy consumption of wireless sensor networks by SystemC, in *Proc. of the Fifth International Conference on Systems and Networks Communications (ICSNC)* (Nice, France, 22–27 August 2010), pp. 94–98
82. W Du, D Navarro, F Mieyeville, Performance evaluation of IEEE 802.15.4 sensor networks in industrial applications. Int. J. Commun. Syst (2014)
83. F Mieyeville, M Ichchou, G Scorletti, D Navarro, W Du, Wireless sensor networks for active vibration control in automobile structures. Smart Mater. Struct. **21**(7) (2012)

84. H Park, W Liao, KH Tam, MB Srivastava, L He, A unified network and node level simulation framework for wireless sensor networks, in *Proc. of the 2009 IEEE Int. Conf. on Communications* (Alaska, USA, 11–15 May 2003)
85. TK Tan, A Raghunathan, NK Jha, EMSIM: an energy simulation framework for an embedded operating system, in *Proc. of the IEEE Int Symposium on Circuits and Systems* (Scottsdale, USA, 26–29 May, 2002), pp. 70–77
86. M Varshney, D Xu, M Srivastava, R Bagrodia, Proc. of the Int. Conf. on Information Processing in Sensor Networks (Cambridge, USA, 25–27 April, 2007)

doi:10.1186/1687-1499-2014-106
Cite this article as: Du *et al.*: Modeling and simulation of networked low-power embedded systems: a taxonomy. *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:106.