Simulation Tools for Mobile Ad-hoc Sensor Networks: A State-of-the-Art Survey

Mohamad Nazim Jambli*, Halikul Lenando*, Kartinah Zen*, Sinarwati Mohamad Suhaili†, Alan Tully‡
*Faculty of Computer Science & Information Technology, Universiti Malaysia Sarawak, Malaysia
†Pre-University Studies, Universiti Malaysia Sarawak, Malaysia
‡School of Computing Science, Newcastle University, UK

***Abstract*—Mobile ad-hoc sensor networks (MASNETs) have recently become an important area of research for Mobile ad-hoc networks (MANETs) researchers. The increasing capabilities and the decreasing costs of sensors make MASNETs applications possible to be deployed in real world. However, before such application can be deployed, their performance need to be measured first. But, the use of real-world measurement is costly and time consuming. Therefore, it is more economical and practical to use simulation tools to simulate MASNETs applications. Although, there are many existing simulation tools for MANETs, most of them are not suitable for MASNETs due to resource-constraint of such networks. Therefore, it is essential to have a survey of the existing simulation tools for MASNETs. In this paper, the comparative study on different simulation tools is conducted to identify the most appropriate tool for MASNETs.**

***Index Terms*—Simulation tools, Network simulator, Mobile wireless sensor networks**

I. Introduction

Mobile wireless sensor networks (MWSNs) have recently become an important area of research for WSN researchers. The increasing capabilities and the decreasing costs of mobile sensors make MWSN applications such as Ocean Monitoring [1], Cattle [2], LISTSENse [3], CenWits [4] and PinPtr [5] become possible and practical to be implemented in real mobile environment. In this type of networks, mobility plays a key role in the deployment of these application. Furthermore, recent studies show that many researchers have proposed mobility-based routing protocols [6] for MWSNs to support mobile applications. Most of these protocols are compared and evaluated through simulation because it is very difficult to duplicate the real world such as in battle field. Thus, the use of real-world measurement is currently almost impossible, certainly costly and time consuming. Therefore, it is more economical and practical to use simulation tools to simulate MWSN applications in mobile environment and to create a statistically significant amount of test runs.

There are many network simulation tools currently available for studying WSNs including GloMoSim, OPNET, ns-2 [7], but choosing the right one for a given MWSN application is very important. To make this choice easier, we conduct a survey of several commonly used simulators that we find significant and interesting. In this paper, we present the results of the comparative study on different WSN simulation tools for evaluating routing protocols in MWSNs. In the second section, we present our methodology and evaluation criteria. In the third section, we describe and compare selected simulation tools according to the methodology and evaluation criteria. In the fourth section, we further discuss our findings, and in the fifth section, we conclude the paper and outlines the future work.

II. Methodology and Criteria

There are many different possible platforms for simulation and evaluating of routing protocols for WSNs. Several studies have been done in comparing different simulators for WSNs as in [8], [9], [10], [7], [11], [12]. But some of the presented information is outdated and most of these studies are not focus on simulation tools for MWSNs where a mobility is one of the important factors that need to be considered. In contrast to that, we make a more detailed survey of WSNs simulators that most appropriate for MWNs. We define a set of criteria to evaluate and compare the selected simulation tools. These criteria can be distinguished into mandatory and optional criteria based on the requirement priority. The mandatory criteria are the main requirements of simulation tools, and it will be more convenient if the simulation tools can also satisfied the optional criteria. The following are the evaluation criteria that have been identified and classified into mandatory and optional criteria.

A. Mandatory Criteria

The mandatory criteria are the main requirements that are essential for any simulation tools to be able to simulate MWSNs. These criteria are evaluated on a yes (✓) / no (✗) basis. Simulation tools that fail to meet all the required criteria are given no further consideration. The mandatory evaluation criteria are designed so that a yes (✓) / no (✗) is easy to determine. There are three mandatory criteria identified for the best selection of MWSN simulation tools as follows:

- WSN-Specific. The selected simulation tools should be designed to simulate specific WSN applications and not just as a general purpose tools. If the tools is not designed as a WSN-Specific simulation tool, it might not consider some unique characteristics of WSN which are needed to accurately simulate the real WSNs environment.
- Energy model. The selected simulation tools need to provide some sort of energy models that able to examine the energy consumption of each static or mobile sensor

978-0-7695-4959-0/13 $25.00 © 2013 IEEE
DOI 10.1109/ACSAT.2012.84

nodes and the whole network when simulating any routing protocols for static or mobile WSNs.

- Mobility model. The selected simulation tools must support some type of mobility models such as Random Way Point, Manhattan, and Gauss Markov [13] and allow user to modify network topology when simulating MWSN programs. They should also be able to examine the accuracy of simulation results when the network topology has been changed in mobile environments.

B. Optional Criteria

The optional criteria are the extra requirements for any simulation tools for MWSNs. These criteria are also evaluated on a yes (✓) / no (✗) basis as in mandatory criteria. It is better for any simulation tools to satify most of these extra criteria for ease of use and get more accurate experimental results for MWSNs. The following are six optional criteria that have been identified.

- Free license - There are various software licenses for simulation tools ranging from very restrictive proprietary licenses to free or open-source licenses. Ideally, these tools must be free, so that they can be easily obtained, used and extended.
- Bridging of code: The simulation tools must bridge the gap between algorithm conception and actual field implementation. They should allow developers to test and verify the code that will run on real hardware with minimum changes. It is even better if they can use the same code in simulation as in real sensor node. For example, they be able to simulate the MWSNs by directly built from TinyOS code.
- Scalability. The simulation tools should be extremely scalable and should be run efficiently in handling large networks (i.e. more than 1000 nodes) in a wide range of configurations.
- Protocols support. The simulation tools must be able to examine separately for each important layer or segment of WSN: radio propagation, physical (PHY) layer, medium access control (MAC) layer, network layer, transport layer and sensing. The lacking of available protocol models in these tools will cause the increase of developing time.
- Technical support: The simulation tools must provide sufficient technical support (i.e. help, documentation, tutorials and maintenance) to help shortening the learning curve and accelerate development process.
- GUI support. Graphical User Interface (GUI) support for simulations can be used as a debugging aid, a visualization and composition tool to view debugging errors or results and facilitates the design of small experiments or the composition of basic modules.

III. Simulation Tools

In this section, the existing simulation tools that are popular and commonly used are evaluated and compared in order to identify the most suitable simulation tool for MWSNs research. The comparison study will be done based on the mandatory and optional criteria defined in the previous section.

There are fifteen existing simulation tools for WSNs have been selected for comparison including NS-2 [14], OPNET Modeler [15], GloMoSim [16], QualNet [17], J-Sim [18], OMNeT++ [19], Castalia [20], SENS[21], SENSE [22], Shawn [23], Avrora[24], TOSSIM [25], ATEMU [26], EmStar [27] and COOJA [28]. The selection of these simulation tools are based on their popularity, interesting characteristics and key features in simulating WSNs. The brief description of these tools are as follows:

- NS-2 (network simulator version two) [14] is a discrete event network simulator targeted at networking research. It provides substantial support for simulation of TCP, routing protocols, and multicast protocols over wired and wireless networks especially in ad-hoc networking research. It was built in C++ and provides a simulation interface through OTcl. It is an open source and licensed for use under version 2 of the GNU.
- OPNET (Optimized Network Engineering Tools) Modeler [15] is a commercialized software tool for network modeling, simulating, analyzing and designing communication networks, devices, protocols, applications. The users can analyze simulated networks to compare the impact of different technology designs on end-to-end behavior. It provides high fidelity modeling, simulation, and analysis of a broad range of wireless networks.
- GloMoSim (Global Mobile Information System Simulator) [16] is a scalable network protocol simulation software that simulates wireless and wired network systems. It is designed using the parallel discrete event simulation capability provided by Parsec, a parallel programming language. It is built using a layered approach to allow the rapid integration of models developed at different layers by different people.
- QualNet [17] is a commercial version of GloMoSim simulator used by Scalable Network Technologies (SNT) for their defense projects. It can predict wireless, wired and mixed platform network and networking device performance. It also can explore and analyze early-stage device designs and application code in closed, synthetic networks at real time speed. It allows users to set up, develop, and run custom network models.
- J-Sim [18] is a discrete event, platform-independent, extensible and reusable Java-based simulation environment for building quantitative numeric models and analyzing them with respect to experimental reference data. It provides GUI library, which facilities users to model or compile the Mathematical Modeling Language. It provides open source models and online documents. In addition, it also can simulate real-time processes.
- OMNeT++ [19] is an extensible, modular, component-based C++ and open-architecture discrete event simulation framework. It is commonly use for simulation of computer networks, but it is also used for queuing

network simulations. It provides the infrastructure for writing such simulations. Specific application areas are catered by various simulation models and frameworks, most of them open source.

- Castalia [20]is a simulator for Wireless Sensor Networks (WSN), Body Area Networks (BAN) and generally networks of low-power embedded devices. It is based on the OMNeT++ platform and can be used by researchers and developers to test their distributed algorithms and protocols in realistic wireless channel and radio models. It can also be used to evaluate different platform characteristics for specific applications.
- SENS[21] is a customizable sensor network simulator for WSN applications, consisting of interchangeable and extensible components for applications, network communication, and the physical environment. It enables realistic simulations, by using values from real sensors to represent the behaviour of component implementation. It allows users to execute the same source code on simulated sensor nodes as deployed on actual sensor nodes.
- SENSE (Sensor Network Simulator and Emulator) [22] is a component-based sensor network simulator written in C++ and developed on top of COST, a general purpose discrete event simulator. It implements sensors as a collection of static components. Connections between each component are in the format of in ports and out ports. This allows for independence between components and enables straightforward extensibility and reusability.
- Shawn [23] is a customizable sensor network simulator based on an algorithmic approach that designed to support large-scale network simulation. The primary design goals of Shawn are: simulate the effect caused by a phenomenon, scalability and support for extremely large networks and free choice of the implementation model. Instead of simulating the effects of a phenomenon, Shawn simulates only the caused effects.
- Avrora[24] is an open-source cycle- accurate simulation and analysis tools for embedded sensing programs written for the AVR microcontroller produced by Atmel and the Mica2 sensor nodes. It contains a flexible framework for simulating and analyzing assembly programs, providing a clean Java API and infrastructure for experimentation, profiling, and analysis. It simulates a network of motes, runs the actual microcontroller programs, and runs accurate simulations of the devices and the radio communication.
- TOSSIM [25] is a discrete event simulator for TinyOS sensor networks that is part of the official TinyOS package developed at UC Berkeley. It captures the behavior and interactions of networks not on the packet level but at network bit granularity. It is designed specifically for TinyOS applications to be run on MICA Motes. It simulates entire TinyOS applications by replacing components with simulation implementations. To compile TinyOS code no additional modifications have to be made to the source code, instead just another make target has do be defined.
- ATEMU [26] is an emulator of an AVR processor used in the MICA platform for WSN which developed in C. It provides GUI to run codes on sensor nodes, debug codes and monitor program executions. It is a specific emulator for WSNs that can support users to run TinyOS on MICA2 hardware. It also can emulate the communication among the sensors and every instruction implemented in each sensor. It provides open sources and online documents.
- EmStar [27] is an emulator specifically designed for WSN built in C, and it was first developed by University of California, Los Angeles. It provides a flexible environment for transitioning between simulation and deployment for iPAQ-class sensor nodes running Linux. Users can run many virtual nodes on a single host with a simulated network, many virtual nodes on a single host with each virtual node bridged to a real-world one for networking, a single real node on a host with a network interface.
- COOJA [29] COOJA is a simulator for the Contiki sensor node operating system. It was originally developed for Cygwin/Windows and Linux platform, but has later been ported to MacOS. It combines low- level simulation of sensor node hardware and simulation of high-level behavior in a single simulation. A simulated Contiki Mote in COOJA is an actual compiled and executing Contiki system.

Firstly, all these fifteen selected simulation tools are reviewed and compared based on the predefined mandatory criteria. These mandatory criteria are very important in order to get more accurate and reliable experimental results for MWSNs evaluation study. The sources of information for this comparison study are basically from scientific papers, vendor web sites and available documentation. Table I shows the comparison study of the selected simulation tools for WSNs based on the mandatory criteria only. The results from this comparative study will be used for further evaluation with optional criteria.

IV. Comparative Study of Selected Simulation Tools

For our comparison study, we have reviewed and identified fifteen potential WSN simulators based on their popularity, interesting characteristics and key features. Based on Table I, there are only five simulation tools out of fifteen that can be considered for further evaluation because of their capability to simulate specific WSN applications and ability to provide energy and mobility models. They are SENSE [22], Avrora[24], TOSSIM [25], EmStar [27] and COOJA [28] simulation tools. These tools will be evaluated again in details based on both mandatory and optional criteria that have been defined earlier.

The following are the details description of the selected simulation tools in terms of their advantages and disadvantages.

TABLE I
COMPARISON STUDY OF EXISTING SIMULATION TOOLS FOR WSNS

Simulation Tool	Latest Version	WSN-Specific Simulation Tool	Provide Energy Model	Support Mobility Model
NS-2 [14]	2.35 (Nov 2011)	✗	✓	✓
OPNET Modeler [15]	17.1 (Dec 2010)	✗	✓	✓
GloMoSim [16]	2.0 (Dec 2000)	✗	✓	✓
QualNet [17]	5.0 (Nov 2009)	✗	✓	✓
J-Sim[18]	2.06 (Feb 2012)	✗	✓	✓
OMNeT++ [19]	4.0 (Mar 2009)	✗	✓	✓
Castalia [20]	3.2 (Mar 2011)	✗	✓	✓
SENS [21]	jan31-2005b (Jan 2005)	✓	✓	✗
SENSE [22]	3.1 (Nov 2008)	✓	✓	✓
Shawn [23]	SVN (May 2010)	✓	✗	✗
Avrora [24]	Beta-1.7.106 (Aug 2008)	✓	✓	✓
TOSSIM [25]	2.1.1 (Apr 2010)	✓	✓	✓
ATEMU [26]	0.4 (Jan 2004)	✓	✓	✗
EmStar [27]	2.5 (Oct 2005)	✓	✓	✓
COOJA [29]	2.4 (July 2010)	✓	✓	✓

A. SENSE

The following are the advantages and disadvantages of SENSE [22], [30].

- Advantages: One of the advantages of SENSE is its balanced consideration of modeling methodology and simulation efficiency. It is a user friendly and fast simulator. It is based on a novel component-oriented simulation methodology that promotes extensibility and reusability to the maximum degree. At the same time, the simulation efficiency and the issue of scalability was considered. It also supports a sufficient energy model and parallelization for WSNs. It can provide different battery models, application, network, MAC and physical layer functionalities. It also integrates G-Sense tool to improve on its ease of use through graphical input of simulation parameters, save and load simulation features, and simulation results management.
- Disadvantages: Although the core of the simulator has been gradually stabilized, SENSE is still in its active development phase. At the moment, it still lacks a comprehensive set of models and a wide variety of configuration templates for WSNs.

B. Avrora

The advantages and disadvantages of Avrora [24] are as follows:

- Advantages: Avrora is an accurate and scalable simulator for the actual hardware platform on which sensor programs run. It provides a framework for program analysis, allowing static checking of embedded software. It is also capable of running a complete sensor network simulation with full timing accuracy, allowing programs to communicate via the radio using the software stack provided in TinyOS [31]. It is language and operating system independence where it can simulate any platforms like Mica2 and MicaZ, and run AVR elf-binary or assembly codes for both platforms. It is implemented in Java unlike TOSSIM [32], which helps flexibility and portability. It can provides a wide range of tools that can be used in simulating WSNs such as control flow graph generation, energy analysis, and the distance-attenuation, and RWP mobility model. Since, it runs code instruction by instruction and avoids synchronizing all nodes after every instruction to achieve better scalability and speed.
- Disadvantages: Although Avrora have many advantages but it also has some drawbacks. Oneof them is that it does not model clock drift, a phenomenon where nodes may run at slightly different clock frequencies over time due to manufacturing tolerances, temperature, and battery performance. It is also does not provide GUI and fifty percent slower than TOSSIM [32].

C. TOSSIM

There are several advantages and disadvantages of TOSSIM [32] as follows:

- Advantages: TOSSIM simulates the TinyOS [31] network stack at the bit level, allowing experimentation with low-level protocols in addition to top-level application systems. It provides several mechanisms for interacting with the network, packet traffic can be monitored and packets can be statically or dynamically injected into the network. It can support thousands of nodes simulation and provide more precise simulation result at component levels because of compiling directly to native codes. This

is a very good feature, because it can more accurately simulate the real world situation. It also has a GUI, TinyViz [32], which is very convenience for the user to interact with electronic devices because it provides images instead of text commands. It also has an add-on PowerTOSSIMz [33] that can be used to measure energy consumption in WSNs.

- Disadvantages: TOSSIM is designed to simulate behaviors and applications of TinyOS, and not to simulate the performance metrics of other new protocols. Therefore, it cannot correctly simulate issues of the energy consumption in WSNs. Moreover, every node has to run on NesC programming language that is implemented on TinyOS. Thus, it can only emulate motes-like nodes and the type of homogeneous applications.

D. EmStar

EmStar [27], [34] have some advantages and disadvantages when people use it to simulate WSNs. They are as follows:

- Advantages: EmStar allows the users to run each module separately without sacrificing the reusability of the software due to its modular programming model. It has a robustness feature that it can mitigate faults among the sensors and evaluate much easier. There is a flexible environment in EmStar that users can freely change between deployment and simulation among sensors. Also with a standard interfaces, each service can easily be interconnected. It also has a GUI, which is very helpful for users to control electronic devices. In addition, it provides many online documents to facilities the widely use of this emulator.
- Disadvantages: However, this emulator contains some disadvantages. For example, it can not support large number of sensors simulation, and the limited scalability will decrease the reality of simulation. In addition, it can only run in real time simulation. Moreover, it can only apply to iPAQ-class sensor nodes and MICA2 motes. All these disadvantages limit the use of this emulator.

E. COOJA

The following are the advantages and disadvantages of COOJA [28] in simulating WSNs.

- Advantages: COOJA is primarily a code level simulator for networks consisting of nodes running Contiki OS. Nodes with different simulated hardware and different on-board software may co-exist in the same simulation. It is flexible and extensible in that all levels of the system can be changed such as sensor node platforms and operating system software. Code level simulation is achieved by compiling Contiki core, user processes and special simulation glue drivers into object code native to the simulator platform, and then executing this object code from COOJA [35]. It is able to execute the Contiki programs either by compiling the program code directly on the host CPU, or compiling it for the MSP430 hardware. It can simulate sensor networks simultaneously at different levels, including the operating system level and the network application level.
- Disadvantages: However, due to its extendibility, the simulator has relatively low efficiency. Simulating many nodes with several interfaces each requires a lot of calculations. It supports a limited number of simultaneous node types, the simulator has to be restarted once and a while if the number of nodes exceed allowable limit. A test interface GUI is absent, thus making extensive and time-dependent simulations difficult.

V. Discussion

Data presented in previous section enable us to compare simulation tools and give rough guidelines for their usage. From the comparison matrix of selected simulation tools for MWSNs (as in Table II), the main contenders for simulating MWSNs appeared to be Avrora and Tossim because these two simulators can simulate mobile environment and at the same time can bridging TinyOS codes into hardware implementation. In addition, these tools also able to provide different models to support MWSN implementation amd can support large number of sensors simulation in comparing to other simlation tools. Although, these simulators are lacking in GUI, but the technical and protocols support provided help users to use these tools effectively.

Based on our experiences using both simulation tools, we have found Avrora more easier to use than TOSSIM. Using Avrora with TinyOS [31] provides a high quality simulation of the actual code that runs on a Micaz node. This allows for more in-depth testing and debugging of applications that do not quite do what they should. Avrora also more flexible and portable than TOSSIM where it can simulates each node as its own thread while still run actual MICA code. Moreover, Avrora can simulate different programming code projects, but TOSSIM can only support TinyOS simulation. Since AVRora simulates the actual AVR chip, custom kernel configurations are not needed in this form of simulation. As such the simulation will take on a form more similar to what is seen in an actual deployment.

VI. Conclusion and Future Work

We have provided comprehensive study on a number of different commonly used simulation tools for WSNs. From the comparison study, we can identified that Avrora is the most appropriate simulation tool for our research works because most of the criteria that we defined for simulating MWSNs can be provided by Avrora. However, some of the simulation tools outweigh others simulation tools in terms of different evaluation criteria. It is suggested that when selecting any simulation tools, it should based on the specific research requirements criteria to support particular research works. It might also useful to integrate different simulators to get a better results. These results will also help WSNs researchers to find out which of the simulation results are the one closest to their works.

TABLE II
COMPARISON MATRIX OF SELECTED SIMULATION TOOLS FOR MWSNS

Evaluation Metric	SENSE [22]	Avrora [24]	TOSSIM [36]	EmStar [27]	COOJA [28]
WSN-Specific	✓	✓	✓	✓	✓
Energy model	✓	✓	✓	✓	✓
Mobility model	✓	✓	✓	✓	✓
Free license	✓	✓	✓	✓	✓
Bridging of code	✗	✓	✓	✓	✓
Scalability	✓	✓	✓	✗	✗
Protocols support	✓	✓	✓	✓	✓
Technical support	✓	✓	✓	✓	✓
GUI support	✗	✗	✗	✓	✗

REFERENCES

[1] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis, "Collective Motion, Sensor Networks, and Ocean Sampling," *Proceedings of the IEEE*, vol. 95, pp. 48–74, Jan. 2007.

[2] T. Wark, P. Corke, P. Sikka, L. Klingbeil, Y. Guo, C. Crossman, P. Valencia, D. Swain, and G. Bishop-Hurley, "Transforming agriculture through pervasive wireless sensor networks," *IEEE Pervasive Computing*, vol. 6, pp. 50–57, Apr. 2007.

[3] C. R. Baker, K. Armijo, S. Belka, M. Benhabib, V. Bhargava, N. Burkhart, A. D. Minassians, G. Dervisoglu, L. Gutnik, M. B. Haick, C. Ho, M. Koplow, J. Mangold, S. Robinson, M. Rosa, M. Schwartz, C. Sims, H. Stoffregen, A. Waterbury, E. S. Leland, T. Pering, and P. K. Wright, "Wireless sensor networks for home health care," in *Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops - Volume 02*, AINAW '07, (Washington, DC, USA), pp. 832–837, IEEE Computer Society, 2007.

[4] J.-H. Huang, S. Amjad, and S. Mishra, "Cenwits: a sensor-based loosely coupled search and rescue system using witnesses," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, (New York, NY, USA), pp. 180–191, ACM, 2005.

[5] G. Simon, M. Maróti, A. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton, "Sensor network-based countersniper system," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, (New York, NY, USA), pp. 1–12, ACM, 2004.

[6] M. S. Shio Kumar Singh and D. Singh, "Routing protocols in wireless sensor networks - a survey," *International Journal of Computer Science and Engineering Survey (IJCSES)*, vol. 1, no. 2, pp. 63 – 83, 2010.

[7] C. Singh, O. Vyas, and M. Tiwari, "A survey of simulation in sensor networks," in *Computational Intelligence for Modelling Control Automation, 2008 International Conference on*, pp. 867 –872, dec. 2008.

[8] M. Imran, A. Said, and H. Hasbullah, "A survey of simulators, emulators and testbeds for wireless sensor networks," in *Information Technology (ITSim), 2010 International Symposium in*, vol. 2, pp. 897 –902, june 2010.

[9] M. Jevtic, N. Zogovic, and G. Dimic, "Evaluation of wireless sensor network simulators," *Proceedings of the 17th Telecommunications Forum TELFOR 2009 Belgrade Serbia*, pp. 1303–1306, 2009.

[10] J. Lessmann, P. Janacik, L. Lachev, and D. Orfanus, "Comparative study of wireless network simulators," *Span*, pp. 517–523, 2008.

[11] D. Curren, "A survey of simulation in sensor networks," *Architecture*, pp. 867–872, 2008.

[12] X. Mao, M. Yang, and D. Mao, "Survey on wireless sensor network applications," *Jisuanji Yingyong yu Ruanjian*, 2008.

[13] A. Mateska and L. Gavrilovska, "An overview of mobility models in wireless sensor networks," in *PhD-NOW* (S. Krco and K. Wrona, eds.), (Sophia Antipolis, France), David Coudert, 2008.

[14] "ns (simulator)." http://www.isi.edu/nsnam/ns/.

[15] "Opnet." http://www.opnet.com/index.html.

[16] "Glomosim - global mobile information systems simulation library."

[17] "Qualnet." http://www.scalable-networks.com/content/products/qualnet.

[18] "Jsim: A sensor, environment and network simulator."

[19] "Omnet++." http://www.omnetpp.org/.

[20] "Castalia - a simulator for wsns." http://castalia.npc.nicta.com.au/.

[21] "Sens: A sensor, environment and network simulator." http://osl.cs.uiuc.edu/sens/.

[22] "Sense - sensor network simulator and emulator." http://www.ita.cs.rpi.edu/.

[23] "Shawn: A customizable sensor network simulator." https://www.itm.uni-luebeck.de/ShawnWiki/index.php/Shawn_Introduction.

[24] "Avrora - the avr simulation and analysis framework." http://compilers.cs.ucla.edu/avrora/.

[25] P. Levis and N. Lee, "Tossim : A simulator for tinyos networks," *UC Berkeley September*, pp. 1–17, 2003.

[26] "atemu - sensor network emulator / simulator / debugger." http://www.hynet.umd.edu/research/atemu/.

[27] "Emstar: Software for wireless sensor networks." http://www.lecs.cs.ucla.edu/emstar/.

[28] F. sterlind and Swedish, *A Sensor Network Simulator for the Contiki OS*. SICS technical report, Swedish institute of computer science, 2006.

[29] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, LCN '04, (Washington, DC, USA), pp. 455–462, IEEE Computer Society, 2004.

[30] G. Chen, J. Branch, M. Pflug, L. Zhu, and B. Szymanski, "Sense: A wireless sensor network simulator," in *Advances in Pervasive Computing and Networking* (B. K. Szymanski and B. Yener, eds.), pp. 249–267, Springer US, 2005.

[31] P. Levis, S. Madden, J. Polastre, R. Szewczyk, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, and D. Culler, "Tinyos: An operating system for sensor networks," in *in Ambient Intelligence*, Springer Verlag, 2004.

[32] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, SenSys '03, (New York, NY, USA), pp. 126–137, ACM, 2003.

[33] E. Perla, A. Cathin, R. S. Carbajo, M. Huggard, and C. Mc Goldrick, "Powertossim z: realistic energy modelling for wireless sensor network environments," in *Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, PM2HW2N '08, (New York, NY, USA), pp. 35–42, ACM, 2008.

[34] J. Elson, S. Bien, N. Busek, V. Bychkovskiy, A. Cerpa, D. Ganesan, L. Girod, B. Greenstein, T. Schoellhammer, T. Stathopoulos, and D. Estrin, "Emstar: An environment for developing wireless embedded systems software," 2003.

[35] N. F. N. T. A. D. T. V. R. S. P. J. M. Joakim Eriksson, Fredrik Österlind, "Cooja/mspsim: interoperability testing for wireless sensor networks," in *SimuTools*, p. 27, 2009.

[36] B. Titzer, D. K. Lee, and J. Palsberg, "Avrora: scalable sensor network simulation with precise timing," in *IPSN*, pp. 477–482, 2005.