2009 Fifth International Conference on Networking and Services

Survey of Wireless Sensor Networks Simulation Tools for Demanding Applications

Marko Korkalainen, Mikko Sallinen, Niilo Kärkkäinen, Pirkka Tukeva
VTT – Technical Research Centre of Finland, Kaitoväylä 1, FI-90570 Oulu, Finland,
Marko.Korkalainen@vtt.fi

Abstract

Requirements for Wireless Sensor networks will increase in the future. Requirements are demanding, especially in industrial real time networks which require high reliability and performance. Network simulation tools are often needed in the network design phase before actual implementation. In this paper, we selected 5 common wireless sensor networks simulation tools and estimated their suitability for high-performance network planning and verification. According to the study, some tools managed the requirements for demanding simulation but needed some extension.

Keywords: simulation, sensor networks, real-time, modeling

1. Introduction

Wireless sensor networks (WSN) are one of the most actively developing areas in network research communities. As the technologies for wireless nodes improve, the requirements for networking are increasing. That enables possibilities for new applications. To reduce costs and time of the deployment process, simulation of the network is a preferred task before testing with real hardware.

In this paper, we review 5 WSN simulation tools to find out how suitable they are for demanding modeling and simulation. Modeling task includes real-time, energy efficiency and routing protocol simulation and accurate radio modeling with 3D definition of the indoor or outdoor environment. The simulator should allow dynamic environment changes with moving sensor nodes or obstacles. To simplify simulation of the software development code, it should be interchangeable between the sensor node platform and the simulator. The target applications are wireless indoor and outdoor control systems and data acquisition networks.

We selected the tools based on the information obtained from the references about the properties of the tools for simulations, popularity of the tool among research communities, the possible extensions for WSN and the active maintenance or support available.

2. Requirements of the Modeling

Simulation of wireless control systems needs detailed modeling of real time properties of the nodes and detailed environment and radio models to obtain realistic enough results. Figure 1 illustrates the sensor node features that we require the simulation tool to model.

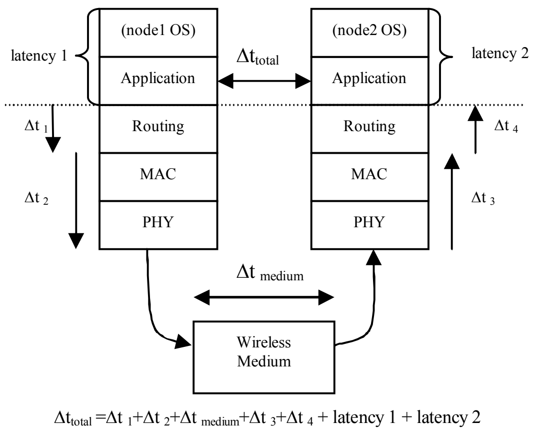

Figure 1. Wireless network layer model

In figure 1, sensor node 1 sends a message to sensor node 2 which receives and processes it after a time delay Δt_{total}. Δt_{total} is the most important quantity since it determines the response time of the network. The latencies between possible node operating systems, e.g., TinyOS [1] and application layers, are built up by software execution in the nodes.

Modeling of the routing layer is needed to evaluate the effect of different routing protocols and network topologies on network operation and delay.

MAC and physical layer model a specific radio system including the delays they produce. An example of this is 802.15.4 type MAC, commonly used for sensor networks, that also defines the physical layer. The physical layer handles data transmission and reception. It is usually interfaced with a wireless medium model that estimates the radio signal strength, quality and delay between the transmitter and receiver unit.

3. Reviewed modeling tools

We made the reviews based on published or publicly available information about the simulation tools. A summary table will be shown at the end of the section.

3.1 NS-2

NS-2 is a very popular general purpose discrete event simulation tool for sensor networks [2]. Currently, NS-2 is actively maintained and used in academic research since it is easily extendable and based on open source.

For mobile sensor networks, the simulator has support for 802.11 and 802.15.4 type wireless MAC. The latter is more suitable for sensor networks since it includes some basic energy modeling and is more close to those used in common sensor nodes.

Sensor node routing algorithm support covers standard wireless IP network ad hoc protocols. The simulator can arrange simple mobility to simulations. More complex mobility can be imported from external applications like BonnMotion and Bonntraffic [3].

NS-2 simulations are written with C++/C and OTCL languages. Protocols and simulation library are written in C++/C, while OTCl works as the control language to create the simulation environment. Simulations can be observed graphically by Network AniMator (NAM). After compiling the simulation source to executable and running it to generate trace files, simulation results can be observed graphically by using Network AniMator (NAM).

Simulations, wireless channel models and omnidirectional antennas are defined in two dimensional free space that excludes detailed environment modeling taking into account the effect of obstacles and structures. Basic sensor node energy modeling is simplified to radio receive, send and idle states. NS-2 has no capability to model real-time OS or application code execution delays. Delays in MAC and wireless channel level can be defined. RT-sim extension for NS-2 implements support for real-time kernel simulations but it is not included in NS-2 release [4].

NS-2 does not have good scalability for large sensor networks since exponential simulation time slowdown. The simulator lacks an application model and is provided with protocols, hardware models and sensing support that are quite different from those used in sensor nodes. There are a great number of public extensions available and the active development process of the simulator raises the risk of bugs and inaccurate or faulty results in simulations. To avoid this, NS-2 maintains a list of verified simulation models on its homepage. [2, 5]

3.2 OMNeT++

OMNeT++ is a public source component-based discrete event network simulator [6]. The simulator mainly supports standard wired and wireless IP communication networks, but some extensions for WSN exist. Like NS-2, OMNeT++ is popular, extensible and actively maintained by its user community in the Academia who has also produced extensions for WSN simulation.

OMNeT++ uses C++ language for simulation models. Simulation models (modules) are assembled with high-level language NED into larger components to represent greater systems. The simulator has graphical tools for simulation building and evaluating results in real time [7].

OMNeT++ is capable of running most TinyOS simulations by NesCT application that converts TinyOS source to simulator compatible C++ code [8]. NesC simulation code interchange between sensor platforms is possible but only in a restricted sense, because the protocol and hardware implementation in the simulator is simplified, and not all hardware is supported. The simulator scales well for very large scale network topologies being limited by memory capacity of the computer used [9]. OMNet++ can not model OS-application layer execution time delays. Delays for lower layers, e.g., MAC and wireless channel, are definable. Without the proper simulation model or framework extensions, the simulator lacks suitable protocols and proper energy modeling for sensor networks, since basic support is mainly for IP networks.

EYES framework for OMNeT++ is written for self-organizing and collaborative energy-efficient sensor networks. EYES enables two-dimensional definition of the simulation map with different failing and error probabilities on different regions of the map caused e.g. by obstacles and fading. [10]

Castalia [11] is the most recent general purpose simulation environment for sensor networks and built on OMNeT++ platform. Castalia is modular and extensible. Its strongest features are accurate wireless channel and radio modeling, including MAC.

3.3 Prowler

Prowler is an event-driven wireless network simulator designed to run in Matlab environment. The simulator, written originally to simulate Berkeley MICA motes, is extendable also for more general platforms. Prowler is implemented in Matlab language (m-file) which makes direct simulation code, e.g., routing protocol or application, interchange between simulator and sensor platforms impossible. Benefits gained from Matlab environment are easy prototyping of applications, integration of different optimization algorithms, GUI interface and good visualization capabilities [12].

The simulator has a deterministic mode that can be used for e.g., application code testing and probabilistic mode for wireless communication channel and sensor node low-level protocol simulations. Prowler scales well to simulate an arbitrary number of sensor nodes with any kind of application, and dynamic topology changes are possible. Prowler is capable of simulating the radio transmission, propagation, reception including collisions in ad hoc radio networks, and the operation of the MAC-layer. The radio models are based on specific signal strength calculations combined with random errors [12]. Modeling support for simulation e.g., application, and radio propagation is provided by m-file plug-ins.

Prowler is well suited for protocol and algorithm development, and it also can provide e.g., QoS metrics on the performance of the tested application. Prowler does not have sensor node energy modeling and needs an extension for that. Also, only one TinyOS MAC protocol has been implemented by default [13]. A publicly available simulator user guide and documentation is unavailable currently. Prowler does not provide several radio propagation models, only two have been implemented. The transceiver wireless channel error models are defined in 2D space, so no 3D enclosed space simulations can be made without modifications. No detailed antenna modeling is included in the Prowler package.

Prowler has good extensibility via plug-ins. After creating modifications to existing simulation models, Prowler has been used for WSN routing protocol energy cost analysis, delivery rate and packet delay analysis [14]. Suitable application areas for Prowler are communication and routing protocol development, debugging, optimization, parameter tuning and arbitrary application prototyping.

3.4 TOSSIM

TinyOS [1] is an open-source operating system specially developed for the wireless embedded sensor networks. There are few hardware platforms available for TinyOS, some commercial and some non-commercial. TinyOS release includes a simulator called TOSSIM. It is built especially for Berkeley Mica Mote platform.

Developers had set four requirements for TOSSIM: scalability, completeness, fidelity and bridging. To be scalable, a simulator should manage networks of thousands of nodes in a wide variety of configurations. To achieve this, each node in TOSSIM is connected in a directed graph where each edge has a probabilistic bit error. For completeness, a simulator must capture behavior and interactions of a system at a wide variety of levels. And for fidelity, a simulator must capture behavior of a network with a subtle timing of interactions on a mote and between motes. Requirement for bridging is met as the simulated code runs directly in a real mote. [15]

TOSSIM is an emulator rather than a simulator, as it runs actual application code. Simulated application code can be transferred directly to the platform, but it might not run in a mote as it runs in a simulation due to the simplifying assumptions in TOSSIM. One inaccuracy source in simulation is TOSSIM's probabilistic bit error model for the wireless medium. This makes TOSSIM efficient and scalable but on the other hand, makes TOSSIM unusable in evaluating low-level protocols. Another simplifying limitation is that every node must run the exact code. While TOSSIM simulates a mote's hardware, including digital I/O, ADC and sensors, it does not model the physical phenomena that are sensed. One drawback in TOSSIM is a lack of energy consumption modeling which is quite important in wireless sensor networks. [15, 5, 16]

When considering high performance real time simulations, one source of error is zero assumed execution time of application code. Only the time instants of events that trigger code execution have been tied to the clock frequency of a sensor node [15].

Also, interrupt handling is simplified (overlapped interrupts are not registered if one is pending).

There are few extensions for TOSSIM. The most important of those are TinyViz, a visualization tool and PowerTOSSIM, an energy consumption modeling add-on [16]. A wireless propagation model and RF physical stack better suitable for IEEE 802.15.4 standard is introduced to increase the accuracy of wireless simulation results [17]. Also, TOSSIM provides communication services for interacting with other applications. External programs can connect to TOSSIM via TCP socket to monitor and actuate to a running simulation [15].

3.5 OPNET

OPNET Modeler is a discrete event, object oriented, general purpose network simulator. Modeler was introduced in 1987 as the first commercial network simulator [18]. Originally, the software was developed for military purposes, but it has grown to be the world's leading commercial network simulation and modeling tool. [18] OPNET is a large and powerful software with a wide variety of possibilities. OPNET can be used as a research tool and also as a network design/analysis tool.

OPNET was originally built for the simulation of fixed networks, and therefore, it contains extensive libraries of accurate models from commercially available fixed network hardware and protocols.

Recent versions also include wide possibilities for wireless network simulations including support for Zigbee compatible 802.15.4 MAC. Strength of OPNET in wireless network simulations is the accurate modeling of the radio transmission. Different characteristics of physical-link transceivers, antennas and antenna patterns are modeled in detail. With Wireless suite for Defence extension OPNET can model 3D outdoor scenarios and take into account different kinds of obstacles like terrain shape and buildings [19]. OPNET can also be used to define custom packet formats. A weak point is that there exists only a few ready models for recent wireless systems. [18, 20].

OPNET uses a hierarchical model to define each aspect of the system. Hierarchical structure is divided into three levels. The top level consists of the project editor, where network topology is designed. The next level is the node level, where individual network nodes and data flow models are defined. A third level is the process editor, which uses a finite state machine approach to support specification of protocols, resources, applications and queuing policies. Finally, a simulation tool is included to support the three higher levels. OPNET also has so-called ESD (External System Domain) for communicating with external software and systems. Via ESD external software can exchange data and influence running simulation in OPNET. [18, 20]

Table 1. Summary about reviewed simulation tools

	NS-2	OMNeT++	Prowler	OPNET	TOSSIM
License	GPL	academic,commercial	academic	commercial	open source
Sensor platform support	universal	universal, (TinyOS)	universal, TinyOS	universal	TinyOS (Berkeley Mica Mote)
Simulation code support	C++,C	C++,C,(nesC)	m-file	C, C++	C,C++,nesC
Simulation code exportable (protocols, etc.)	limited	limited, (NesC yes)	no	limited	yes
Scalability to large networks (n>100)	fair (some cases)	good	excellent	excellent	excellent
Protocol design/opimization	possible	possible	possible	possible	possible
Mobile network simulation	yes	yes	yes	yes	no
Dynamic network topotology change possible	yes	yes	yes	yes	no
Real-time OS/SW execution time modelling	no	no	no	no	limited
Needs extensions for proper WSN simulation	yes	yes	yes	yes	yes
Custom extensions can be written	yes	yes	yes	yes	yes
3D radio modelling	no	no	no	yes	no
Mobile network simulation	yes	yes	yes	yes	no
Standard MAC and routing support	802.11, 802.15.4, DSDV, DSR, TORA, AODV	802.11, fair (ad hoc routing)	802.15.4, fair (ad hoc routing)	802.11, 802.16, UMTS, SMART MAC, 802.15.4. AODV, DSR, GRP, OLSR, OSPFv3, TORA	CSMA, primitive
Standard energy modelling (battery/radio)	primitive / TX, RX, idle	primitive / TX, RX, sleep	primitive	primitive / TX, idle, sleep	none
Publicly available extensions for WSN	energy models, protocols, SensorSim	energy models, protocols, SensorSim, EWSNSIM, Castalia	Plugins for energy models, protocols	Several under license	TinyViz, PowerTOSSIM
Implemented radio models	shadowing, 2-ray ground, free space	free space, 2-ray ground (experimental, Castalia)	probabilistic, deterministic	free space, CCIR, Hata, Longley-Rice, TIREM, Walfish-Ikegami	probabilistic bit error

4. Analysis and conclusions

Five commonly used WSN modeling tools were presented and a summary table about their features was shown, see table 1.

We found that the presented WSN simulators, e.g., NS-2 and OMNeT++, need to be extended or modified for more accurate WSN simulation. Default-supplied protocols, energy models, physical layer and environment models are too simplified or of the wrong type. Also many of the simulators are developed for a specific modelling task in which they are accurate and appropriate (e.g., only a certain type of routing protocol, MAC, energy or physical model has been implemented).

All reviewed simulators allow custom extensions to be written or downloaded from Internet for better WSN modeling. Commercial simulator OPNET, offers better support, maintenance and proven simulation models.

Very high performance real time simulations that include operating system and application layer code execution delays are impossible with the presented simulators without extensions or use of external software.

Mobile network simulations and dynamic topology changes are possible in all of the simulators excluding TOSSIM. When considering scalability to large networks NS-2 lags behind the other reviewed simulators.

Simulation code exportation, e.g., C/C++ or TinyOS routing algorithm or application, to sensor node is easiest from commercial platform dependent TinyOS emulators, e.g., TOSSIM since they run most application code used on sensor nodes. In general purpose platform independent simulators, e.g., NS-2 C/C++, source code interchange is possible when required interfaces and libraries are implemented on the target platform and the simulator.

All presented modeling tools lack detailed environment modeling. Excluding the OPNET radio models are defined in two-dimensional space. None of the presented simulators is capable of 3D indoor simulation without modifications.

5. References

[1] TinyOS, Available from: http://www.tinyos.net

[2] The Network Simulator – ns-2, Available from: http://www.isi.edu/nsnam/ns

[3] BonnMotion mobility framework, Available from: http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/

[4] Pagano P, Prashant Batra, Lipari G, “ A Framework for Modeling Operating System Mechanisms in the Simulation of Network Protocols for Real-Time Distributed Systems”, Parallel and Distributed Processing Symposium, 2007.

[5] David Curren, “A Survey of Simulation in Sensor Networks”, Univ. of Binghamton, bj92489@binghamton.edu

[6] C. Mallanda, A. Suri, V. Kunchakarra, S.S. Iyengar, R. Kannan, and A. Durresi, ”Simulating Wireless Sensor Networks with OMNeT++”

[7] Omnest network simulation, Available http://www.omnest.com/network-simulation.php

[8] [Omnet++ Discrete event simulation system, Available from: http://www.omnetpp.org/,

[9] http://ctieware.eng.monash.edu.au/twiki/bin/view/Simulation/OMNeTppComparison

[10] EYES WSN Simulation Framework, Available from: http:// www.ses.cs.utwente.nl/ewsnsim

[11] Castalia: A Simulator, Available from: http://castalia.npc.nicta.com.au/

[12] Prowler network simulator, Available from: http://www.isis.vanderbilt.edu/Projects/nest/prowler

[13] Thomas W. Carley, “Sidh: A Wireless Sensor Network Simulator”, University of Maryland at College Park

[14] Guoliang Xing, Chenyang Lu, Robert Pless , “Minimum Power Configuration in Wireless Sensor Networks”

[15] Levis P., Lee N., “TOSSIM: A Simulator for TinyOS Networks”, pal@cs.berkeley.edu, September 17, 2003

[16] Shnayder V., Hempstead M., Bor-rong C., Allen G.W., Welsh M. (2004), “Simulating the Power Consumption of Large-Scale Sensor Network Applications”, SenSys ’04, November 3-5, 2004, Baltimore, Maryland, USA.

[17] Suh C., Joung J-E., Ko Y-B. (2007), “New RF Models of the TinyOS Simulator for IEEE 802.15.4 Standard”, IEEE Communications Society, WCNC 2007 proceedings.

[18] http://www.opnet.com/solutions/brochures/Modeler.pdf (Read 5.2.2008)

[19]http://www.opnet.com/solutions/network_rd/modeler_wi reless_defense.html (Read 10.2.2008)

[20] Prokkola, J.(2006), “OPNET – Network simulator”, VTT Technical Research Center of Finland