SFB 637 Autonomous Logistics

Becker, M.; Timm-Giel, A.; Murray, K.; Lynch, C.; Görg, C.; Pesch, D.: Comparative Simulations of WSN. In: Cunningham, P.; Cunningham, M. (eds.): ICT-MobileSummit 2008. 2008

Comparative Simulations of WSN

Andreas TIMM-GIEL[1], Ken MURRAY[2], Markus BECKER[1], Ciaran LYNCH[2], Carmelita GÖRG[1], Dirk PESCH[2]

[1]*TZI, University of Bremen, Otto-Hahn-Allee NW 1, 28359 Bremen, Germany*
Tel: +49 421 218 9719, Email: [atg, mab, goerg]@tzi.de
[2]*Cork Institute of Technology, Rossa Avenue, Cork, Ireland*
Tel: +353 21 4326668, Email: [ken.murray, ciaran.lynch, dpesch]@cit.ie

Abstract: In this paper we present the results of a study within the CRUISE Network of Excellence. Within CRUISE we have simulated a simple scenario for Wireless Sensor Networks with three different simulation tools: NS-2, OMNeT++ and OPNET. The scenario investigated is that of a fire fighter entering a building and deploying sensor nodes in different rooms. Data collected from the sensor nodes is transmitted to the fire fighter and the incident commander at the other end. The simulation tools are compared regarding their ease of implementing the scenarios, collecting the metrics delay, throughput as well as comparability of the results.

Keywords: Wireless Sensor Networks, Simulation Modeling

1. Introduction

A broad variety of different simulation tools are used to simulate key characteristics of Wireless Sensor Networks. They range from emulator originated tools like Avrora and TOSSIM to wireless and mobile communication simulation environments, like OMNeT++, OPNET and NS-2. Each of these classes and tools has its specific advantages and disadvantages and often the selection of the tool is mainly based on the experience of the researcher rather than on rational arguments.

An overview of the different tools and simulation environments with their particular pros and cons has been established by the CRUISE project [1] and is given in [2]. The next step within the CRUISE WP *Software Tools for modeling, design and simulation* is to compare the tools using an identical simulation scenario that can be easily implemented in the different simulation environments, i.e. having a rather common mobility model, a commonly used network layer protocol etc.

The authors with a background in Communication Networks also tried to simulate the scenario using the emulators TOSSIM and Avrora. However, there was more time required to fully understand the simulators themselves, to implement the reference scenario and add or adopt the required functionality in the emulators. The results will be published in future work of the authors.

The rest of the paper is structured as follows: in the next section the three different simulation environments are briefly introduced. This is followed by a detailed description of the fire fighter application scenario and the configuration in each of the model layers. For each of the layers the restrictions and particularities of the different simulation environments are given. The simulation results for each of the metrics are discussed and finally the paper ends with an outlook and conclusions.

2. Simulation Tools

The following provides a brief introduction to the three simulation tools investigated within this study.

2.1 – OMNeT++

The Objective Module Network Test-bed in C++ (OMNeT++) [3] is a component-based, modular and open-architecture simulation environment with strong GUI support and an embeddable simulation kernel. OMNeT++ provides a hierarchical nested architecture.

The modules are programmed in C++, the GUI of OMNeT++ is created using the Tk library. The modules are assembled into components and models by using a high-level language (NED). Modules communicate by sending messages. The simulation configuration is managed by .ini files.

Today there are several sensor network simulation frameworks based on OMNeT++. The Mobility Framework [4] implements the support for node mobility, dynamic connection management and a wireless channel model. Currently the Mobility Framework provides only models for IEEE 802.11. A module for 802.15.4 has been developed based on the Mobility Framework, but is not available to the public yet [5]. An implementation of AODV exists as an extension of the INET framework.

2.2 – NS-2

NS-2 [6] is a discrete-event simulator written in C++ with a TCL front-end, intended for networking research. It is free and open source, but it is not supported commercially. Development of the simulator is ongoing on the current NS-2, as well as on the follow-up simulation tool NS-3.

As with all discrete-event simulators, precise timing simulation (i.e. of code execution) is not possible, although a timing model can be added into the simulation. NS-2 uses TCL for scenario generation – this allows complex scenarios to be generated automatically by scripts. The simulator is controlled by TCL commands.

Originally, NS-2 only supported simulation of fixed TCP/IP based computer networks. Mobile nodes are however now supported to allow the simulation of mobile ad-hoc networks. Ad-hoc routing protocols supported by NS-2 are AODV, DSDV, DSR and TORA. Mobility simulation however required an extension, since the standard NS-2 simulation is based on the idea of fixed links between interfaces, which are no longer static in wireless scenarios. Mixing wired and wireless nodes in the same NS-2 simulation is also difficult.

In general, nodes in NS-2 are considerably more sophisticated than the typical sensor node. Layers are included in the models that are not practical in a sensor node implementation, and the presence of these layers is likely to distort a simulation. For example, all mobile nodes include packet queues at each interface, have unlimited packet storage, have a unique address (such as an IP address) and run ARP to resolve addresses. The 802.11 DCF MAC protocol is implemented for the MAC. A wide range of fixed routing protocols, transport protocols and application models (such as web services) are provided however these are likely to be of little use for a sensor network simulation.

Propagation models supported are free-space, two-ray ground reflection and shadowing. Simple energy modeling is supported; this tracks the energy used for each packet transmitted and received. Contributed models provide support for other protocols (e.g. IEEE 802.11 and IEEE 802.15.4). These are not part of the core distribution and are still in development. Overall, sensor network simulation is not easily supported by NS-2 although many researchers are currently attempting to modify NS-2 towards better WSN simulation.

2.3 – OPNET

The OPNET Modeler is a commercial network simulation software by Opnet Technologies, Inc. A free academic license is available [7]. A Graphical User Interface support the

configuration of the scenarios and the development of network models. Three hierarchical levels for configuration are differentiated: The network level creating the topology of the network under investigation, the node level defining the behaviour of the node and controlling the flow of data between different functional elements inside the node, and the process level, describing the underlying protocols, are represented by finite state machines (FSMs) and are created with states and transitions between states. The source code is based on C/C++. The analysis of simulated data is supported by a variety of built-in functions. Different graphical presentations for the simulation results exist.

OPNET develops specialized modules like Wireless, UMTS, etc. Additional modules are contributed by the University Program. For mobility models random waypoint, arbitrary trajectories, and mobility updates from external sources via the HLA (Higher Layer Architecture) are supported. For modelling the radio propagation, OPNET provides CCIR, Free Space, Hata, Longley-Rice, TIREM, or Walfish-Ikegami. Additionally Rayleigh, Ricean or Two-Ray models are available from the OPNET community, i.e. not supported by OPNET. 802.15.4/ZigBee model is under development, a non supported version is available [8]. Energy models are not directly supported by OPNET.

3. Application Scenario

The application investigated is taken from the IST wearIT@work project [9]. The wearIT@work project is developing a set of new solutions to support the mobile workers of the future. These solutions are based on wearable computing technology and their effectiveness and applicability are being tested on four different pilot studies in the fields of Healthcare, Emergency Rescue, Aircraft Maintenance and Production Management and Training. Within the emergency management field, it is imperative that fire fighters have a good communication link to the command post as this can directly impact the survival of himself or a victim. The concept of a virtual sensor network lifeline is investigated. The sensor nodes of this lifeline are deployed by the fire fighter when entering the building, e.g. by a mechanism in the fire fighters boot. The sensor nodes can measure temperature and possibly detect smoke (gas) in the environment and inform the firefighter as well as the incident commander on the other side. Additionally it can be used to exchange status information and voice messages between the firefighter and the incident commander.

4. Specification of Scenario

The fire fighter scenario under investigation is depicted in Figure 1. Node 0 is the fire fighter; node 1 is the Incident Commander. Nodes 2 to 25 are sensing nodes deployed by the firefighter when entering the building. The scenario is depicted in Figure 1. All nodes except for the firefighter node are at fixed locations. The firefighter node moves with a speed of 0.5 km/h to the south until it reaches the bend, from there it moves in western direction to the end of the hallway. The nodes 0 and 1 are active from the start of the simulation; the nodes 2…25 are enabled when the firefighter moves by.

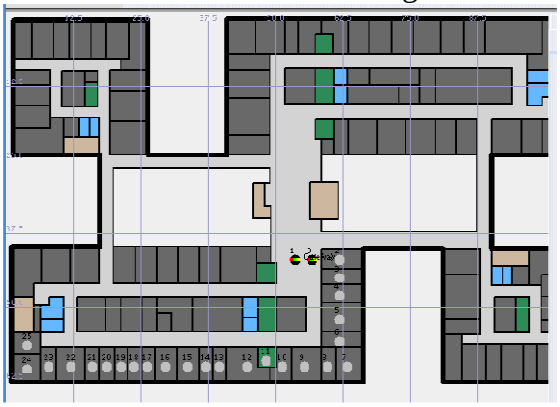

Figure 1. Firefighter Scenario

Within the application layer, the fire fighter node starts unicasting data packets to the gateway at the simulation start time. The nodes 2…25 are alternately transmitting to the firefighter (node 0) and the gateway (node 1), the firefighter and the gateway are sending only to each other. The rate of sending packets is 0.2/s (i.e. one packet every 5s). The size of the packet is limited by the lower layers (no fragmentation). Currently we are employing a packet size of 32 bytes.

For comparison reasons the AODV routing protocol is used, as it is implemented for most simulation environments. For sensor networks and sensor nodes with TinyOS and IEEE 802.15.4 physical and MAC layer usually TinyAODV is implemented. TinyAODV is available with TOSSIM and Avrora. OPNET and NS-2 use the full AODV as specified in IETF the MANET working group [10], therefore differences in the results are to be expected. Flooding as a network layer protocol was used within OMNeT++, as the interworking of the INET and Mobility Framework could not be guaranteed.

Energy efficient data transmission over sensor networks requires the use of energy efficient MAC protocols. The transmission of beacon packets between transmitter and receiver facilitates low duty cycle in which devices transmissions are coordinated. With this strategy, devices can sleep between the coordinated transmissions, which results in energy efficiency and prolonged network lifetimes. The IEEE 802.15.4 MAC standard for low duty cycle, low data rate devices is the most significant commercially adopted MAC protocol to date [11]. We therefore focus on the use of IEEE 802.15.4 for this comparison of simulation tools.

For this first comparison a simple radio propagation model is used: The nodes within a distance of 10 meters have a perfect channel without packet loss, the nodes more than 10 meters apart cannot communicate.

In order to evaluate the simulation model functionality and performance, metrics are collected at each node in terms of data throughput, packet loss and delay. Where required, the additional tools to process and plot data are presented.

5. Results

The usability and performance of the most commonly used wireless sensor network simulation tools are measured by means of th application scenario as discussed in section 4. The simulation model results will now be presented for each simulation tool investigated in this study.

5.1 – OMNeT++

The propagation/connectivity model mentioned in Section 4 has been implemented, an 802.11 link layer model and a simple Flooding network layer model is used for the simulations done with OMNeT++.

The results produced by OMNeT++ are contained in an output vector file. This file has been post-processed using command-line tools and finally Matlab was used for creating the statistics and figures. So, there is an extra effort needed to present the results.

The throughput and delay received at the firefighter node is depicted in Figure 2. Due to only two nodes (the firefighter and incident commander) being active in the beginning of the simulation the traffic is very low (~0.2 packets/s) initially. With the nodes being switched on one by one, the traffic increases to approximately 5 packets/s at the end of the simulation at 500 s model time. With the increase in traffic there is also an increase in the delay involved (especially as the messages are flooded in the network and the new nodes increase the length of the lifeline). The delay starts out negligible and later on reaches value up to 0.2 seconds.

The throughput and delay received at the command post node is depicted in Figure 3. The values and the behavior remain the same as for the firefighter.

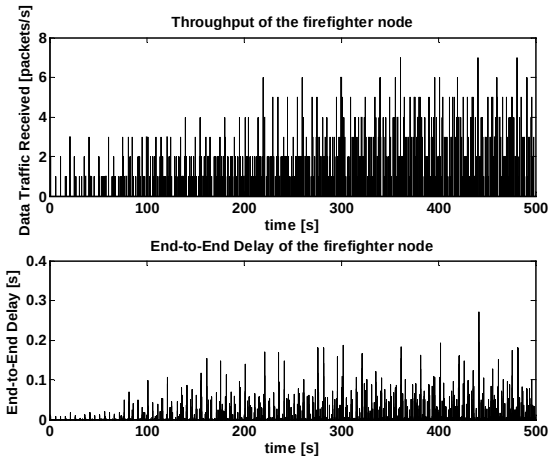

Figure 2. Throughput and delay at firefighter

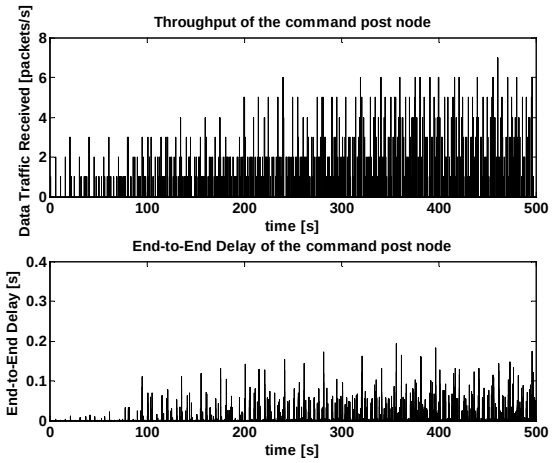

Figure 3. Throughput and delay at command post node

The delay distribution is shown in Figure 4. It can be seen that more than 38 packets reached the destination within 0.02 seconds. Average and Variance of throughput and delay obtained by this OMNeT++ simulation are given in Table 1 for comparison.

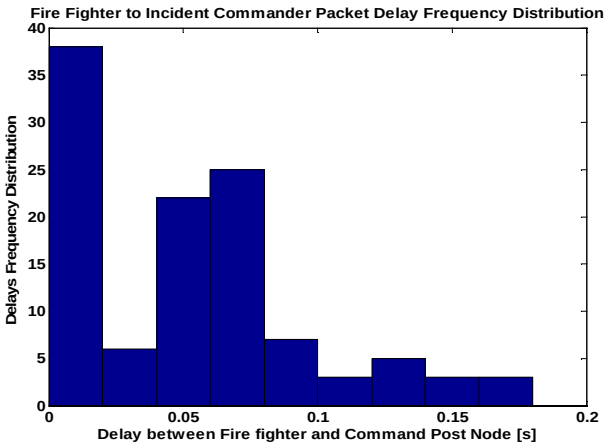

Figure 4. Delay distribution

5.2 – NS-2

As described in section 2.3, the simulation parameters in NS-2 including node mobility and radio propagation are controlled via a user defined TCL script. Using the fire fighter scenario specification a TCL script was created to support the required functionality including node positions, mobility, data traffic profile and initiation. The separation of simulation parameters and scenario definition from the simulated protocols enables the user to quickly and efficiently build scenarios and analyse protocol performance without the need to recode NS-2 modules. The key performance metrics investigated include data throughput at both fire fighter and incident command post nodes, packet delay and frequency distribution and the packet drop rate due to collisions. The most common method of analysing NS-2 trace files is by the development of user defined scripts to extract key performance indicators. This can be one of NS-2's most limiting factors as results cannot be readily analysed and graphed following a simulation. There is however a NS-2 graphing tool called Trace Graph freely available to extract the most commonly used metrics [12]. Trace Graph was used in this study to extract the NS-2 performance metrics.

One of the first performance metrics collected is the data throughput at both the mobile fire fighter and fixed incident command post nodes. Referring to the scenario specification in section 4, as the fire fighter moves he enacts sensor devices which in turn begin transmitting data to both him and the incident command post. Furthermore, from the simulation start time the fire fighter node unicasts data packets to the incident command post. We therefore expect the throughput to increase at both nodes as the simulation progresses. The received throughput at both the fire fighter and incident commander nodes are shown in Figure 5 and Figure 6 respectively. Statistical analysis indicates that the received throughput at the fire fighter and command post are 1.636 packets/second and 1.59 packets/second respectively.

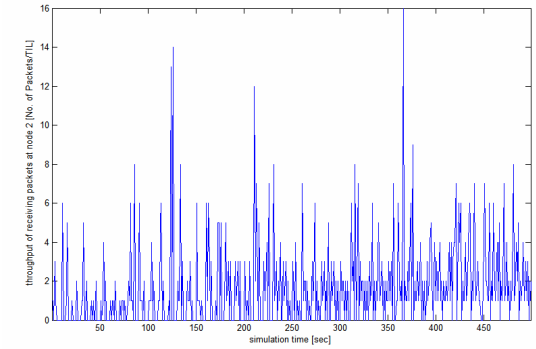

Figure 5 Received throughput at fire fighter node

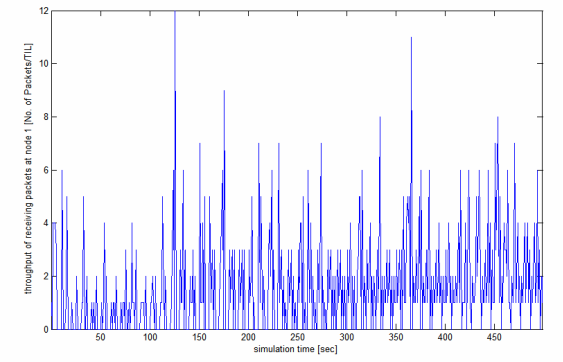

Figure 6 Received throughput at incident commander node

As the fire fighter moves, data packets are transmitted to the incident commander from the fire fighter at a rate of 0.2/s. In an emergency response scenario, it is critical that minimal delay occur in this data transfer. Figure 7 depicts the packet delay along this data path, while Figure 8 shows the packet delay frequency distribution. As Figure 8 indicates, the packet delay from firefighter to command post is less extremely low. An increase in this metric would indicate the requirement for additional gateway devices to be installed along the data path.

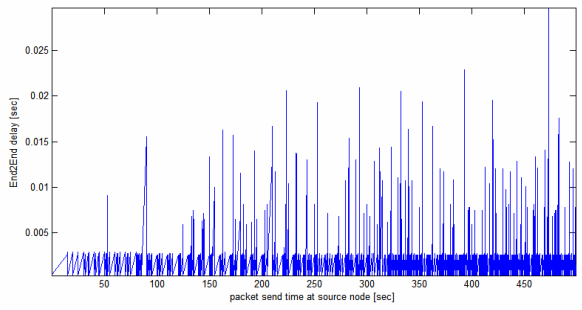

Figure 7 Firefighter to incident commander packet delay

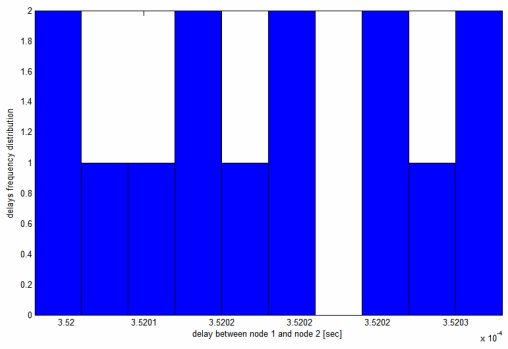

Figure 8 Firefighter to incident commander packet delay frequency distribution

5.3 – OPNET

The scenario was implemented in the OPNET simulator, version 11.5.A PL3 with Wireless 11.5.A package. The wireless sensor node model was created with the help of an available MANET station model and is depicted in Figure 9. Results were obtained in order to analyze the performance of the given network scenario. Comparing the total amount of received acknowledged transmission packets, it can be noticed that the received traffic at the command post is less than that of the firefighter node. That is because not all traffic from the firefighter node reaches the command post. For this mobile scenario it is necessary

to enable active route expiration timers for every node, though that will increase the global network load and will consume more energy. The use of expiration timers is not the most efficient way of improvement, because in this scenario all routes are static except those including the firefighter node.

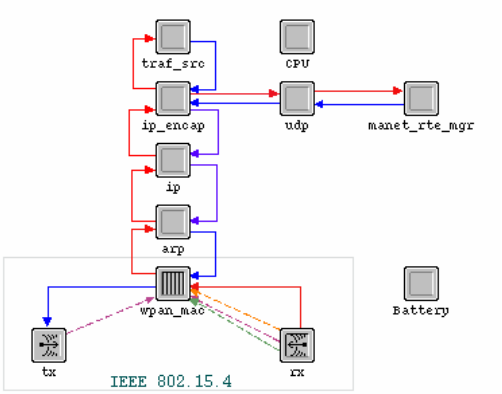

Figure 9 Wireless sensor node model in OPNET

Future work may have the scope on implementation of link failure indication from MAC layer to the Network layer. The other, more complex way is to create a custom AODV model from the existing standard model in OPNET and integrate a network layer control channel for link error indication, i.e. HELLO messages.

Figure 10 depicts the received throughput and delay at the fire fighter node obtained using OPNET. Again the traffic and delay start very low and increase over the duration of the simulation. The delay is similar to the results for OMNeT++ bounded to 0.2 seconds, although different routing algorithms are used in both simulations.

The same performance metrics are shown in Figure 11 for the incident commander. The results obtained are similar to those for the firefighter.

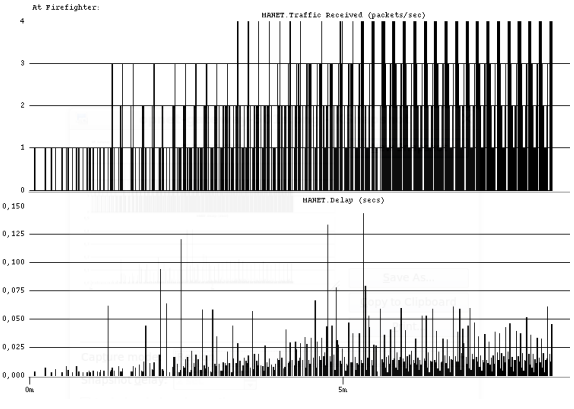

Figure 10 Received throughput and delay at fire fighter node

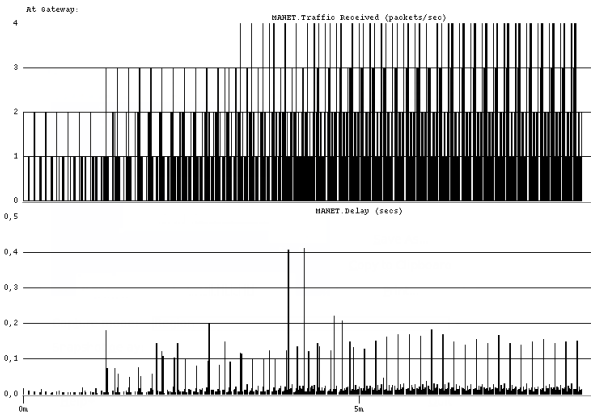

Figure 11 Received throughput and delay at Incident Commander node

6. Platform Comparison

Results are obtained at the command post, including the traffic from all nodes within the simulated environment. Table 1 summarizes key performance statistics at the incident command post. As expected there are differences however, results gathered from each simulator are of the same order of magnitude, despite using different models.

The mean received throughput varies from 1.193 packet/s for OMNeT++, to 1.658 packet/s for OPNET with NS-2 representing very similar results to OPNET at 1.59 packet/s. The mean of the received end-to-end delay at the command post from all stations varies by 20 ms from OMNeT++ to OPNET. The result of 2.3 ms in Table 1 represents the

end-to-end delay between the fire fighter and command post nodes, as it is not possible using the NS2 Tracegraph tool to determine the end-to-end delay from all nodes to the command post. The variances could not be gathered with the Tracegraph tool for NS-2. The variances for the throughput and the delay of OPNET and OMNeT++ are very similar.

Table 1. Comparison of performance results at the incident command post node

	OMNeT++	OPNET	NS-2
Mean of received throughput [packet/s]	1.193	1.658	1.59
Variance of received throughput	0.179	0.1681	-
Mean of received eed [s]	0.051	0.031	0.0023
Variance of received eed	0.002	0.0026	-

7. Conclusions

The exercise of implementing the same scenarios in different simulators has again proven the difficulty of this task due to the different simulation analysis capabilities and protocol support. The simple scenario investigated could not be implemented in any of the simulators without adding functionality or libraries or by using similar instead of the same models. So there is no publicly available IEEE 802.15.4 model for OMNeT++ leading to the need to take 802.11 instead. Also there was no routing algorithm which could be used without major integration work in all three simulators. Here again a different model (flooding instead of AODV) was taken for OMNeT++. These differences (802.11 instead of 802.15.4 and flooding instead of AODV) also explain the significant quantitative differences of the results of the OMNeT++ simulator. OMNeT++ with flooding has significant less traffic throughput than OPNET and ns-2.

The usability of the three simulators cannot be compared - it is a question of taste and experience mainly. OPNET has a convincing analysis and way of presenting the results, however requires significant time to learn to use it. Ns-2 having the charme of being an open source software suffers from bad presentation of results and means to analyse them, as external tools have to be used or be developed. OMNeT++ as an open source simulation tool features a nice GUI, but however has not incorporated a tool for graphical representation of the results.

References

[1] CRUISE Network of Excellence Web Portal, http://www.ist-cruise.eu, last accessed October 2nd, 2007.
[2] http://www.ist-cruise.eu/cruise/Public%20documents/wp123-wsn-simulation-tool-knowledgebase/, last accessed April 1st, 2007
[3] András Varga: The OMNeT++ Discrete Event Simulation System. In the Proceedings of the European Simulation Multiconference (ESM'2001). June 6-9, 2001. Prague, Czech Republic.
[4] Witold Drytkiewicz, Steffen Sroka, Vlado Handziski, Andreas Köpke, Holger Karl: A Mobility Framework for OMNeT++. 3rd International OMNeT++ Workshop, at Budapest University of Technology and Economics, Department of Telecommunications Budapest, Hungary, January 2003.
[5] Feng Chen, Falko Dressler: A Simulation Model of IEEE 802.15.4 in OMNeT++. 6. Fachgespräch "Drahtlose Sensornetze" der GI/ITG-Fachgruppe "Kommunikation und Verteilte Systeme". Aachen, July 2007.
[6] The network simulator - ns-2. http://www.isi.edu/nsnam/ns/, 2002.
[7] http://www.opnet.com, last accessed April 10th, 2007
[8] http://www.open-zb.net, last accessed April 10th, 2007
[9] http://www.wearitatwork.com/ last accessed October 1st, 2007.
[10] Charles E. Perkins, Elizabeth M. Belding-Royer, Samir R. Das: Ad hoc On-Demand Distance Vector (AODV) Routing, RFC 3561. Internet Engineering Task Force, July 2003.
[11] IEEE 802.15.4 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), 2006.
[12] http://www.tracegraph.com, last accessed October 1st, 2007.