International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 5 – No. 9, July 2013 – www.ijais.org

Wireless Ad hoc Network Simulators: Analysis of Characterstic Features, Scalability, Effectiveness and Limitations

Anita Sethi
Uttarakhand Technical University Dehradoon

J. P. Saini
[2] MMEC
Gorakhpur

Manoj Bisht
[3] WWIL
Delhi

ABSTRACT
The nature of wireless ad hoc network is such that communication between the nodes is extremely volatile, resultantly the quality of the wireless links is heavily unstable. Implementation of algorithms with enormous processing power and memory capacity for small and restricted resource-constrained wireless nodes is an inspiring task. Performance and behaviour estimation and comparison of wireless ad hoc network protocols, simulations are upright solution between cost and complexity, on the one side, and accuracy of the results, on the other side. Such simulation tools should allow researchers to verify new ideas and compare the proposed solutions in a virtual environment helping to avoid unnecessary, time-consuming or expensive hardware implementations. Difficulty arises in selection of wireless ad hoc network simulator for specific objective, features and usability due to the prensence of lots of simulators. Reusability, availability, performance, scalability, support for rich-semantics, scripting languages and graphical support are the main requirments for best framework of the wireless ad hoc network simulator. Within the study, we have described desirable and valuable missing features of the simulators, demarcation of their strength and weaknesses. Lot of effort is required for installation, habituation, implementation and visualization of wireless ad hoc network simulator. A lot of researchers and Scholars can seek help from this paper to identify the suitability of simulator for their study.

Keywords
Wireless ad hoc network, scalability, simulator, emulator

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. Testbed implementation generate the most accurate results but required to acquire hardware, the severely restricted monitoring, debugging possibilities, as well as, excessive effort required to create a simulated environment approaching the real application scenario. Solution of Testbed implementation is to capture the reality models up to a limited extent, which implies that simulation results will generally not be as accurate as real implementations. Running a network simulation fundamentally involve implementation of a protocol, designing a network topology and traffic scenario and visualization and analysis of simulation results. The conclusions of this study will permit a quick yet deep enough understanding of features, efficiency, extendibility, accuracy, and easiness of use, which can help other researchers to identify the wireless network simulator that is most suitable for their needs.

2. REQUIREMENTS OF THE MODELING
To model analytically a Wireless ad hoc network is complex and impractical which leads to oversimplified analysis with restricted assurance and deploying testbeds require huge effort. Simulation provides appropriate model based on hardware and physical layer assumptions and a suitable framework to ease implementation. The dynamics of the physical parameters sensed by the network govern the network traffic and topology and non-rechargeable batteries leads to oversimplified analysis with restricted assurance of network operation. The correctness of the model and suitability of a particular tool to implement the model should be evaluated before conducting experiments. Figure 1 illustrates the sensor node features that we require the simulation tool to model. In figure 1, sensor node 1 sends a message to sensor node 2 which receives and processes it after a time delay Δ_{ttotal}. Δ_{ttotal} is the most important quantity since it determines the response time of the network. The latencies between possible node operating systems, e.g., TinyOS 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.

The models contain novel components, not present in classical network simulators, as detailed power and energy consumption models or environment models. Nodes, Environment, Radio channel, Sink nodes and Agents are major components considered in network model of the wireless ad hoc network simulator. Node behavior depends on interacting factors that cause cross-layer interdependencies. Protocol-tier, physical-node tier and media-tier are the

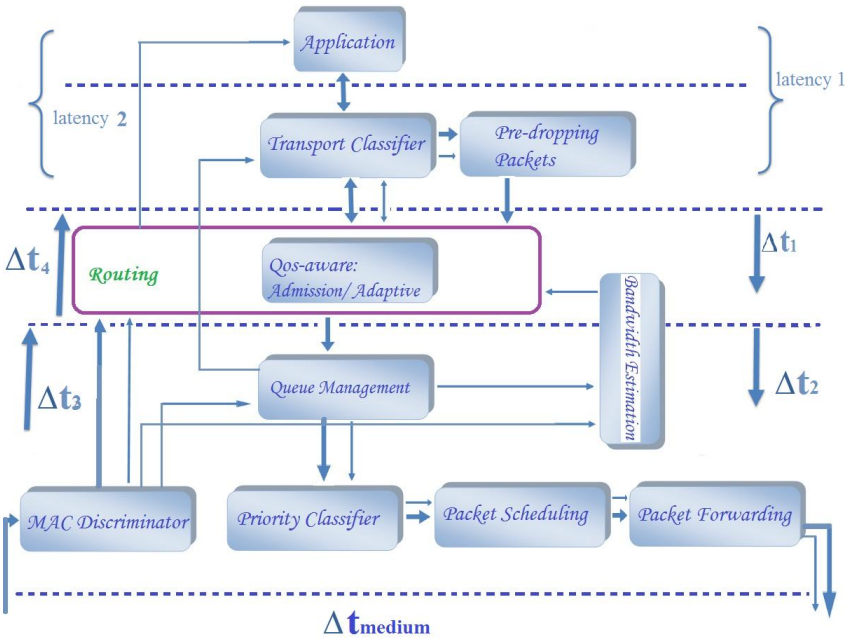

Figure 1: Framestructure of Wireless ad hoc Network Simulator

abstract tiers of the node model. Protocol-tier comprises of MAC layer, a routing layer and a specific application layer whose operation depends on the state of the physical tier. Physical-node tier common elements contains are the set of physical sensors, the energy module and the mobility module.

A node is connected with the environment through a radio channel in media-tier. The selection of a simulation framework for any type of network is a task that is worth to spend enough time. Indeed, this is particularly true for wireless sensors nets, because of the diversity and complexity of the simulation scenarios, protocols, and elements involved. In such a heterogeneous scope, different evaluation tools achieve different goals. This section identifies and discusses the main features to be considered in the selection of a WSN simulation framework.

3. CLASSIFICATION CRITERIA

Algorithm level Simulators emphasize on the data structure, logic and semantics of the algorithms. Localization, distributed routing, flooding based protocols of wireless ad hoc network are analyzed in AlgoSensim. Effect caused by a phenomenon, improve scalability and support free choice of the implementation model are the objectives of Shawn. Message passing view of the network is supported in Sinalgo. Data link and physical layers features are modeled in Packet level Simulators. Battery, radio propagation and sensor channel models are provided in SensorSim, extension of ns-2. Loosely-coupled, component-based programming model, and real-time process-driven simulation is adopted in J-Sim. The parallel discrete event simulation capability is designed in GloMoSim. Instruction level simulator model the CPU execution at the level of instructions or even cycles, called emulators. Atemu is an emulator that can run nodes with distinct applications at the same time. Avrova is a Java-based emulator used for programs written for the AVR microcontroller produced by Atmel.

4. RELATED WORK

Emulators and code level simulators emulate the sensor hardware or process the provided program code in a manner it would be executed on a real device. Command-line framework capable of simulating and analyzing programs in Avrora with a set of monitoring and profiling utilities and Control flow graphs, the graphical representation of executed instructions, may prove to be useful as well. System architecture of the Freemote Emulator defines the Physical, Data Link (MAC), Routing and Application based on IEEE 802.15.4 standard and developed in Java. MSPsim is an emulator in which the program simulates and displays a visual representation of the whole sensor board equipped with elements such as sensors, interfaces and LEDs. The environment of TOSSIM simulates networks at the bit level, so, for hundreds of simulated nodes may communicate with a number of actual nodes and create a common topology running exactly the same TinyOS applications. VMNet compromises power consumption and response time performance metrics, calculated from logs gathered during the simulation. A sensor node platform emulator called WSim supports Timing and interrupt data, as well as memory and power consumption, can be estimated during simulation allowing developers to perform thorough analyses and evaluations.

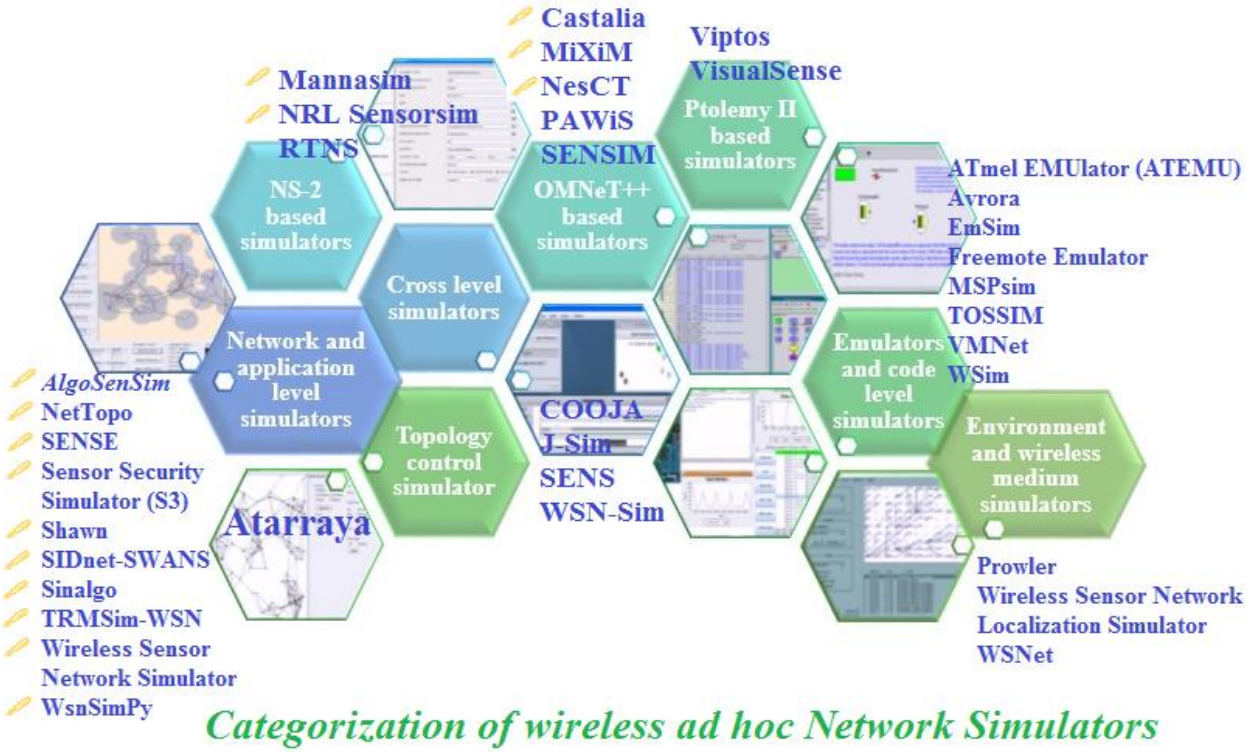

Figure 2

Topology construction and topology maintenance are two processes of Atarraya used for observing various solutions by comparing performance and efficiency in an environment providing a pre-defined energy consumption and communication models. Hierarchical structures of models formed in Ptolemy II based simulators contains Software components, called actors, execute simultaneously, exchanging messages through interconnected ports. Viptos is based on TOSSIM and Ptolemy II provides a low-level simulation of TinyOS programs supports experiments concerning heterogeneous networks where distinct nodes may use different models at each level of simulation. The framework of VisualSense consists of models representing communication channel and nodes with mobility, power management, packet losses, collisions, etc. of both Hierarchical and heterogeneous wireless ad hoc network. Ad hoc On-Demand Distance Vector (AODV), Directed Diffusion, Destination-Sequenced Distance-Vector Routing (DSDV), Dynamically Controlled Routing (DSR), Low-Energy Adaptive Clustering Hierarchy (LEACH) and Temporally Ordered Routing Algorithm (TORA) can be simulated with Mannasim, ns-2 based simulator. NRL Sensorsim simulates and detect parameters of carbon monoxide concentration, seismic activity or audible sound of wireless sensors and power utilization parameters may be described with the use of energy consumption model. Real Time Network Simulator (RTNS) is aimed at providing real time CPU simulations by the integration of the Ns-2 environment.

C++ and NED language is used in Castalia, MiXiM, NesCT, PAWiS, and SENSIM are OMNeT++ based simulators. SENSIM provides the feature of developing new sensor networks protocols and investigate networking and scalability. Radio model defines the antenna power requirements and battery model is used to control the amount of the remaining energy and processor model provides different levels of energy consumption in various states of operation in SENSIM. Concurrent simulation of the power consumption of every sensor is most crucial functionality of PAWiS. TinyOS applications can run in the OMNet++ environment using NesCT simulator. CHannel SIMulator, Mac Emulator, Mobility Framework and Positif Framework is designed in MiXiM comprises of environment, connectivity and mobility, reception and collision, experiment support and protocol library. Development of distributed algorithms or protocols and cross-level analysis is provided in Castalia.

5. REVIEW OF WIRELESS AD HOC NETWORK SIMULATORS

A usage–specific analysis about all the contemporary simulators have been done in the next page. It discusses about key features and limitation of the Simulators. This comparison is crucial to understand the applicability of simulator in different situation.

	Summary	Environment	Simulation language	Key features	Limitations
NS-2	NS-2 is a discrete event simulator targeted at networking research has a modular approach and hence is effectively extensible.	Simulate TCP, routing, and multicast protocols over wired and wireless networks. The simulator focuses on following the ISO/OSI model.	C++ and OTcl.	NS-2's extensibility has an object-oriented apprach which allows for straightforward creation and use of new protocols. Sensor channels, battery models, lightweight protocol stacks, hybrid simulation support, and scenario generation tools are the key features. It provides a visualization tool called NAM (Network AniMator).	Requires advanced skills to perform meaningful and repeatable simulations. Lack of customization available. Packet formats, energy models, MAC protocols, and the sensing hardware models all differ from those found in most sensors. NS-2 also lacks an application model.
TOSSIM	TOSSIM is a discrete event simulator for TinyOS, captures the behavior and interactions of networks not on the packet level but at network bit granularity. TOSSIM works by replacing components with simulation implementations.	TinyOS is a sensor network operating system that runs on custom 'mote' hardware provides accuracy and complexity of the radio model necessary for their simulations.	nesC	TOSSIM simulates the TinyOS network stack at the bit level, allowing experimentation with low-level protocols in addition to top-level application systems. The simulation provides several mechanisms for interacting with the network, packet traffic can be monitored and packets can be statically or dynamically injected into the network. The transmission is simulated at the bit level.	execution model does not capture CPU time Compilation steps lose the fine-grained timing and interrupt properties of the code. Less flexible due to exact same code of each node. TOSSIM does not model energy consumption, though there is an add-on PowerTOSSIM that corrects this problem.
GLoMoSim	Global Mobile Information System Simulator is a scalable simulation environment for large wireless and wired communication networks. The node aggregation technique is introduced into GloMoSim and each node represents a geographical area of the simulation.	GloMoSim supports radio propagation, CSMA MAC protocols, mobile wireless routing protocols, and implementations of UDP and TCP.	PARSEC (Parallel Simulation Environment for Complex Systems)	The ability to use GloMoSim in a parallel environment distinguishes it from most other sensor network simulators. GloMoSim can be executed using a variety of synchronization protocols and was successfully implemented on both shared memory and distributed memory computers.	GloMoSim does not simulate a wired as well as a hybrid network with both wired and wireless capabilities. GloMoSim suffers the same problems as Ns-2, the packet formats, energy models, and MAC protocols are not representative of those used in wireless sensor networks.
UWSim	UWSim, focus on handling scenarios specific to Underwater environments, for example low bandwidth, low frequency, high transmission power, and limited memory. It is based on a network component-based approach, rather than a layer/protocol-based approach.	Tailor-designed for simulations of under-water sensor networks.	object-oriented C#	UWSN needs a simulator which simulates the acoustic network is based on a novel routing protocol proposed by the developers, unlike traditional simulators which are based on either proactive or reactive routing protocols such as AODV and DSR. The various characteristics of underwater networks such as low bandwidth, need for high frequency and the effect of salinity and temperature with depth are considered while simulating sensor networks.	Limited functionalities **UWSim** calls for further extensions to support a wide range of UWSN simulation exercises. **UWSim** cannot be used for simulating any other sensor network except UWSN.
JSim	J-Sim is a general purpose simulator uses the concept of components, replacing the notion that each node should be represented as an object. J-Sim uses three top level components: the target node, the sensor node, and the sink node. Each component is broken into different parts and modeled differently within the simulator which makes it easy to use different protocols in different simulation runs.	J-Sim provides support for sensors and physical phenomena. Energy modeling, with the exception of radio energy consumption, is also appropriately provided for sensor networks.	Java and Jacl, a Java version of Tcl.	J-Sim feature of component based architecture scales better than the object oriented model used by Ns-2 and other simulators. Furthermore, J-Sim features an improved energy model and the ability to simulate the use of sensors for phenomena detection. Like SensorSim, applications may be simulated, and there is support for the connection of real hardware sensors to the simulator.	Relatively complicated to use. Inefficiencies. Unnecessary overhead in the intercommunication model. The only MAC protocol that can be used is 802.11, a problem that seems to occur in most sensor simulators that are built on top of all-purpose simulators.
SENS: A Sensor Environment and Network Simulator	SENS model an application, network communication, and the physical environment. It enables realistic simulations, by using values from real sensors to represent the behavior includes sound and radio signal strength of component and power usage implementations.	Multiple component implementations in customizable sensor network simulator, SENS offer varying degrees of realism. The same source code that is executed on simulated sensor nodes in SENS may also be deployed on actual sensor nodes, this enables application portability.	C++.	Portable applications Novel mechanism for modeling physical environments Tight integration of computation, communication and interaction with the physical environment. SENS defines an environment as a grid of interchangeable tiles.	Less customizable to change the MAC protocol and other low level network protocols. SENS supports only single phenomenon of detectable sound.
COOJA (COntiki Os JAva)	COOJA is a simulator for the Contiki sensor node operating system. MSPSim can be integrated into COOJA, forming COOJA/MSPSim. It allows simultaneous cross-level simulation at application, operating system and machine code instruction set level. COOJA combines low level simulation of sensor node hardware and simulation of high-level behavior in a single simulation.	COOJA is flexible and extensible in that all levels of the system can be changed or replaced: sensor node platforms, operating system software, radio transceivers, and radio propagation models. As network communication is central to a WSN, the COOJA Simulator supports adding and using different radio mediums [28].	Java	Support different simulated hardware and different on-board software in the same simulation. 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. It can simulate sensor networks simultaneously at different levels, including the operating system level and the network (application) level.	Low efficiency. Lot of calculations, 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.

	Summary	Environment	Simulation language	Key features	Limitations
SENSE	SENSE was a sensor network simulator developed in 2004.	SENSE supports an energy model that is sufficient for wireless sensor networks.	CompC++, a component extension to C++.	The most significant feature of fast and user- friendly simulator SENSE is its balanced consideration of modeling methodology and simulation efficiency. A novel component-oriented simulation methodology of SENSE promotes extensibility and reusability to thefast and maximum degree with simulation efficiency and the scalability.	Lacks a comprehensive set of models and a wide variety of configuration templates for wireless sensor networks. Besides, a visualization tool is desirable which can quickly track down what goes wrong during the simulation.
VisualSense	representation and analysis of communication channels, sensors, ad-hoc networking protocols, localization strategies, MAC protocols, energy consumption in sensor nodes, etc. are modeled using component-based in VisualSense,	VisualSense provides an accurate and extensible sound model and radio model, based on a general energy propagation model that can be reused for physical phenomena.	VisualSense Ptolemy II, consists of a few new Java classes and some XML files.	It supports actor-oriented definition of network nodes, wireless communication channels, physical media such as acoustic channels, and wired subsystems. The software architecture consists of a set of base classes for defining channels and sensor nodes, a library of subclasses that provide certain specific channel models and node models, and an extensible visualization framework. It is intended to enable the research community to share models of disjoint aspects of the sensor nets problem and to build models that include sophisticated elements from several aspects.	VisualSense does not provide any protocols above the wireless medium, or any sensor or physical phenomena other than sound.
(J)Prowler	Prowler and (J) Prowler are an event-driven probabilistic wireless sensor network simulator that can be set to operate in either deterministic mode or in probabilistic mode. It can incorporate arbitrary number of motes, on arbitrary (possibly dynamic) topology, and it was designed so that it can easily be embedded into optimization algorithms.	(J)Prowler is targeted to the Berkeley MICA Mote hardware platform running application built on TinyOS, though it could be modified to simulate more general systems	Prowler is written in Matlab, while JProwler is written in Java.	With fast, nice visualization capabilities and easy way to prototype applications,the network simulator models the important aspects of all levels of the communication channel. A simplified and accurate model is used to describe the operation of the Medium Access Control with the nondeterministic nature of the radio propagation is characterized by a probabilistic radio channel model.	(J)Prowler provides an accurate radio model and it provides only one default MAC protocol of TinyOS
Shawn	Shawn is an open source customizable sensor network simulator designed to support large-scale network simulation. It is claimed to provide the highest abstract level and support larger network comparing to other simulators such as ns-2, SENSE, OmNeT++, GloMoSim, and TOSSIM.	The simulation environment is the home of the virtual world in which the simulation objects reside. The simulated nodes reside in a single World instance. The nodes themselves serve as a container for so-called processors.	Java.	Shawn features persistence and decoupling of the simulation environment by introducing the concept of Tags. They attach both persistent and volatile data to individual nodes and the world. They decouple state variables from member variables, thus allowing for an easy implementation of persistence. Another benefit is that parts of a potentially complicated protocol can be replaced without modifying.	Detailed simulations of issues such as radio propagation properties or low-layer issues are not well considered.
Castalia	Castalia is an application-level simulator evaluate different platform characteristics for specific applications for WSN based on OMNeT++. In Castalia, sensor nodes are implemented as compound modules, consisting of sub-modules that represent, for instance, network stack layers, application, and sensor.	It is a generic simulator with realistic wireless channel and radio model based on measured data.	C++	Features of Castalia include: flexible physical process modeling, sensing device bias and noise, node clock drift, and several highly tunable MAC and routing protocols implemented. Distinct physical process modules in Castalia represent different sensing devices (e.g. temperature, pressure, light and acceleration).	It should be noted that Castalia is not sensor-platform specific. Castalia is meant to provide a generic reliable and realistic framework for the first order validation of an algorithm before moving to implementation on a specific sensor platform. It is not useful if one would like to test code compiled for a specific sensor node platform.

5. ACKNOWLEDGMENTS

Our thanks to the experts who have contributed towards development of the template.

6. CONCLUSIONS AND OPEN ISSUES

Simulation is an energetic tool to observe Wireless ad hoc Networks due to the impracticality of analysis and the difficulties of setting up real experiments. This artical provides benchmarks to help selecting a suitable simulation model for a Wireless ad hoc Network and a comprehensive description of the most used available tools. Our primary vision was to prepare an analysis that would put together and compile the most valuable information about wireless ad hoc network simulators. Besides adopting frameworks designed exclusively for WSNs simulations, some technologists have succeeded in making use of general purpose simulators in their sensor networks studies. Subsequently the networking community has less proficiency in the wireless domain than with wired networks, electing abstractions there is even extra challenging. Simulations which shortage essential features can result in ambiguous or inappropriate solutions. Technologists must select their level of simulation detail with caution.

Most of the Simulators use different approaches to solve different problem and hence are incomplete as they do not provide solution for all kinds of problems. The diversity of existing simulation tools has led to accuracy and authenticity issues. Comparison and imitation of evaluation results from competing simulation systems gets more difficult due to such variety of simulators.. Amongst other drawbacks include lack of conception, Graphical User Interface, lack of documentation and lack of illustrations. The predicament of precise, scalable and cost-effective simulation solution can be solved with the use of mixed-mode simulation as an effective midway solution. Mixed mode simulation enables simulation of algorithms partially in software and partially in WSN testbed real hardware. It integrates both simulated environment and a real test bed to improve both the precision and scalabilty of test results

Networking algorithms get strained in similar ways by randon error and as also by detailed model when the networking algorithm is strong to a range of errors and the error is not corelated. Visualisation techniques can help figure out wrong details and control overload. Improvement of correctness and precision can be done by reserchers by making the simulation code available for other reserchers also and description of their simulation setup. Using only real testbeds are not feasible due to their excessive cost and intricacy. Our view point says that theoritical validation of algorithm can serve as a good means for evaluating many algorithms. Further it is observed that the trend in the Wireless Adhoc Networkss field is to use mixed-mode simulation as a provisional solution.

7. REFERENCES

[1] A. Boulis, “Castalia – A simulator for Wireless Sensor Networks and Body Area Networks,” User's Manual, October 2009. accessed June 2012.

[2] C. P. Singh, O. P. Vyas, and M. K. Tiwari, “A Survey of Simulation in Sensor Networks,” in Proceedings of CIMCA'08, 2008 International Conference on Computational Intelligence for Modeling Control and Automation, (Vienna, Austria), 10–12 December 2008.

[3] M. Jevti´c, N. Zogovi´c, and G. Dimi´c, “Evaluation of Wireless Sensor Network Simulators,” in Proceedings of TELFOR 2009, 17th Telecommunications forum, (Belgrade, Serbia), 24–16 November 2009.

[4] A. Kellner, K. Behrends, and D. Hogrefe, “Simulation Environments for Wireless Sensor Networks,” tech. rep., Institute of Computer Science, Georg-August-Universit¨at G¨ottingen, June 2010.

[5] S. Mahlknecht, S. A. Madani, and J. Kazm, “Wireless Sensor Networks: Modelling and Simulation, Discrete Event Simulations,” InTech, 2010.

[6] K. Garg, A. F¨orster, D. Puccinelli, and S. Giordano, “Towards Realistic and Credible Wireless Sensor Network Evaluation,” in Proceedings of ADHOCNETS 2011, 3rd International ICST Ad Hoc Networks Conference, (Paris, France), 21–23 September 2011.

[7] H. Sundani, H. Li, V. K. Devabhaktuni, M. Alam, and P. Bhattacharya, “Wireless Sensor Network Simulators A Survey and Comparisons,” International Journal of Computer Networks, vol. 2, pp. 249–256, February 2011.

[8] M. Imran, A. M. Said, and H. Hasbullah, “A Survey of Simulators, Emulators and Testbeds for Wireless Sensor Networks,” in Proceedings of ITSim 2010, 4th International Symposium on Information Technology, vol. 2, (Kuala Lumpur, Malaysia), pp. 897–902, 15–17 June 2010.

[9] “TinyOS operating system.” http://www.tinyos.net. accessed June 2012.

[10] B. Titzer, D. Lee, and J. Palsberg, “Avrora: Scalable Sensor Network Simulation with Precise Timing,” in Proceedings of IPSN'05, Fourth International Conference on Information Processing in Sensor Networks, (Los Angeles, USA), 25–27 April 2005.

[11] B. Titzer and J. Palsberg, “Nonintrusive precision instrumentation of microcontroller software,” in Proceedings of LCTES'05, Conference on Languages, Compilers and Tools for Embedded Systems, (Chicago, USA), 15–17 June 2005.

[12] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin, “EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks,” in Proceedings of General Track: 2004 USENIX Annual Technical Conference, (Boston, USA), 27 June – 2 July 2004.

[13] IEEE Computer Society, “Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs),” in IEEE Standard 802.15.4-2006, Part 15.4, (New York, USA), 8 September 2006.

[14] “WirelessHART Overview.” http://www.hartcomm.org/protocol/wihart/wireless_overview.html. accessed June 2012.