Architecture of the Simulator of the Personal Local Wireless Networks: Examples of implementation

Oleksandr Tymchenko[a], Bohdana Havrysh[b], Mariya Nazarkevych[b], Oleksandr O. Tymchenko[c] and Orest Khamula[c]

[a] University of Warmia and Mazury Olsztyn, Poland
[b] Lviv Polytechnic National University, Lviv, Ukraine
[c] Ukrainian Academy of Printing, Lviv, Ukraine

Abstract

The use of existing wireless network simulators requires the large number of preset simulation settings and node settings. In the case of wireless sensor networks, a large number of node interaction protocols, methods for constructing a communication graph and ensuring a given network connectivity are also used. Therefore, it is advisable in the simulation to abstract to a certain level and not to consider the physical and hardware levels of network nodes, which will reduce the time spent on the study of network parameters. The architecture and graphical interface of the developed real-time simulator "SNOW" for research of models and methods of local networks wireless access construction are considered in the work. The performance indicators of the simulator for modeling the topology control, construction of the communication graph, its visualization and determination of the power consumption parameters of the K-NEIGH type sensor network for variants with sequential and parallel execution of simulation steps are given.

Keywords

Sensor network, construction methods, simulator, simulator architecture, communication graph

1. Introduction

One of the main tasks that developers of sensor and specialized networks face is to ensure the scalability and the necessary parameters of reliability, durability and network performance. This is difficult to achieve without the prior study and analysis of the proposed algorithm characteristics.

Research and evaluation of algorithms and protocols for BSM – the wireless sensor networks can be done in three ways [1, 2]:

- analytical method – the most difficult way due to the large number of influencing factors;
- modeling on real equipment – the most expensive way, you need ready-made equipment and high time costs when conducting experiments;
- computer simulation - the best way due to the development of computing capabilities. There are several ways to model and simulate a complex system.

Including:

- creation and further simulation of a complete model of the system, all its components and connections between them, together with the negative phenomena that may occur during the operation of a real network;

IntelITSIS’2021: 2nd International Workshop on Intelligent Information Technologies and Systems of Information Security, March 24–26, 2021, Khmelnytskyi, Ukraine
EMAIL: olexandr.tymchenko@uwm.edu.pl (O. Tymchenko); dana.havrysh@gmail.com (B. Havrysh); mar.nazarkevych@gmail.com (M. Nazarkevych); olexandr.tymch@gmail.com (O. O. Tymchenko); khamula@gmail.com (O. Khamula)
ORCID: 0000-0001-6315-9375 (O. Tymchenko); 0000-0003-3213-9747 (B. Havrysh); 0000-0002-6528-9867 (M. Nazarkevych); 0000-0003-2774-2138 (O. O. Tymchenko); 0000-0001-7596-0813 (O. Khamula)

- creation and subsequent simulation of a model containing stochastic elements. It is assumed to use random variables to model the consequences of the negative phenomenon, rather than the negative phenomenon itself (path losses, packet delays, etc).

The first option allows you to simulate the behavior of each of the network nodes in detail over time, as well as to simulate the movement of packets in detail and their routing.

The second option is optimal for obtaining some general characteristics of the network, such as the connectivity of the communication graph, the average number of neighbors for each node. This approach is often used to model topology control methods. To determine the general characteristics of a network consisting of a large number of nodes (typical for wireless sensor networks), the second method of modeling is used. Although both the first and the second methods will give the same result for a large number of elements. The fact is that while using the full model, the modeling time increases exponentially with increasing number of elements. It is hundreds of times higher than the time cost of statistical modeling, which has little dependence on changes in the number of network elements by a thousand elements or more [3-5].

2. Related works

In order to select tools for modeling methods for constructing sensor networks, an assessment of existing software products was conducted. Figure 1 shows a diagram that allows to compare the means for simulation on the implemented level abstraction (y-axis) and the maximum possible size of the simulated network (x-axis).

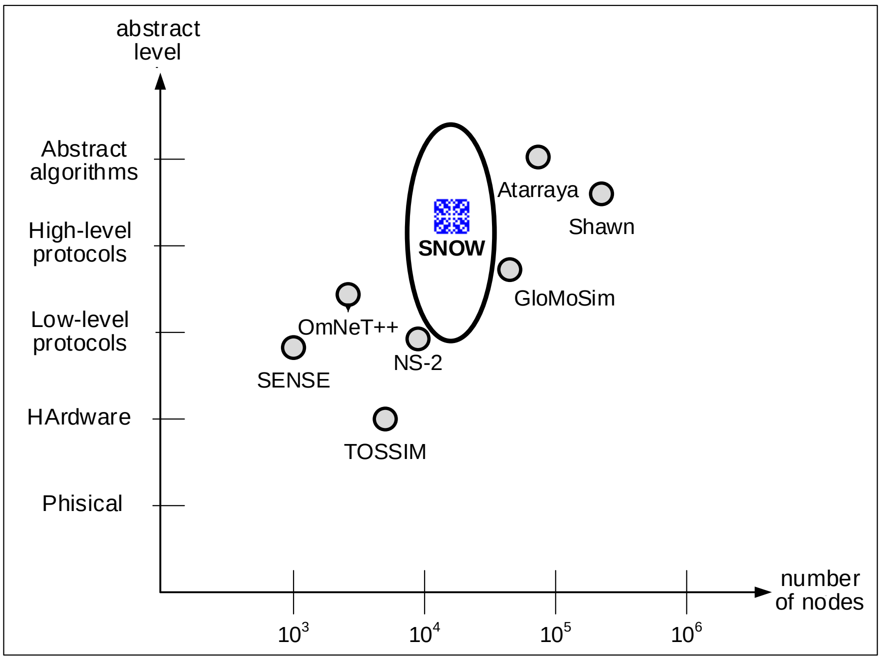

Figure 1: Existing means of CM simulation

In [6, 8], the NS-2 simulator is used to model the functionality of sensor fences for personal access. NS-2 (Network Simulator version 2) is a time-discrete simulator developed at the University of California, Berkeley. NS-2 allows local networks and WAN modeling and supports the detailed modeling of TCP and UDP protocols, routing in networks with both wired and wireless access.

The NS development began in 1989 is constantly being improved. Purpose of NS is education and research in network technologies.

The simulator NS-2 has a basic model that implements the IEEE 802.15.4 standard. For ad-hoc hemming in NS-2, routing protocols AODV, DSDV, DSR and TORA are adopted. They provide an additional support for securing the flexibility of robots with mobile universities. At the same time, only the routing protocols can be used in the NS-2, as it is not up to the point to break the special features of mouthless sensor fences (Figure 2).

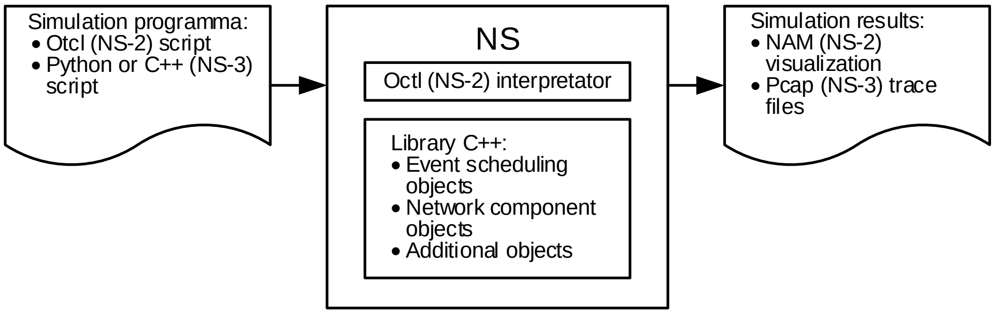

Figure 2: The basic structure of NS-2 and NS-3 simulators

The NS-3 simulator, which is described in [10-13] is much better for stimulating wireless sensor networks. NS-3 (version 3) is a completely excellent simulator based on NS-2 [7-9]. The main difference from NS-2 is the absence of OTcl (short for MIT Object Tcl), the use of programming exclusively in C ++ and Python (Figure 2).

NS uses two programming languages because it has two types of operations to perform. On the one hand, detailed data exchange protocol simulation requires a programming language that can efficiently manipulate bytes, packet headers, and at the same time must enable the implementation of algorithms that work with large data sets. This task requires a high execution speed. Ensuring low time costs of the development cycle (running the simulation, finding an error in the code, error correction, recompilation, re-simulation) are less important.

On the other hand, a large amount of research in the field of network technology requires minor changes in parameters, changes in network configuration or a quick review of possible scenarios. In this case, the above-mentioned iterative development process is more important, and the speed of execution does not play a role. Note that in case of both static and mobile sensor network nodes, it is necessary to investigate not only the network configuration, but also the ways of data transmission, which is quite difficult in this simulator due to the limited graphical interface.

Shawn simulator described in [14] is a program-simulator of discrete events for large wireless sensor networks modeling algorithms. Shawn does not provide the same level of modeling detail as, for example, NS-2, but with the correct construction of the model it gives a convergent result. The Shawn simulator has a very wide range of possibilities for statistical modeling, however does not allow modeling of a phenomenon, but simulates the impact of such. For example, you can simulate the interference of individual packets using a signal propagation model and abstractly set the channel losses proportional to the number of nodes in the transmitter area.

Also, Shawn simulator requires writing your own processors for wireless network nodes, different models for messaging, and more. The modular architecture of the simulator allows additions that are standard for simulators of such types. An alternative approach to the simulation process itself provides high performance.

TOSSIM (TinyOS) – a system specifically designed for sensor networks [14, 15]. It has a software model component described in nesC. TinyOS is not an operating system in the traditional sense. It is a software environment for embedded systems and has a set of components that allow you to create simulation models for a specific application, such as TOSSIM. The TOSSIM simulator can simulate networks of up to several thousand nodes, and by analyzing them, predict the behavior of the network with high accuracy. By modeling networks with possible interferences and errors, the simulator creates a simple but at the same time effective model of various interactions of nodes in the network. Describing a low-power model of TinyOS devices, it simulates the behavior of the sensor node with a high probability, describing its characteristics and conducting a large number of experiments. For the convenience of developers, TOSSIM supports a graphical user interface, providing detailed visualization and reproduction of the actions of the running simulation model, but does not reproduce the communication graph tied to the environment.

There are also other publicly available network simulators, such as JavaSim, SSFNet, Glomosim and Qualnet, in which the developers tried to solve the shortcomings of these systems. JavaSim developers realized the disadvantage of using object-oriented system design and tried to build a component-oriented architecture. However, the effectiveness of the simulation was limited by the choice of the Java simulation language [15, 17].

The SENSE simulator is designed as an efficient and powerful sensor network simulator [15, 16]. It uses a component port model, which frees simulations from the interdependence that is common in object-oriented architecture. The component port model makes simulation models extensible – a new component can replace an old one if they have compatible interfaces, and advanced users have the ability to develop new simulation mechanisms. Removing the interdependence between models also promotes reusability. A component designed for one simulation can be used in another if it meets the requirements of the latter in terms of interface and semantics. In SENSE, there is a level of reusability that has been made possible by the widespread use of the C ++ template: a component is usually declared as a template class so that it can process other types of data. However, SENSE can only use the parallel simulation mechanism for compatible components. Therefore, only in the case of sequential simulation can each component in the model repository be reused.

3. Architecture and of the Simulator of the Personal Local Wireless Networks

The analysis of existing software products for simulation of construction methods and control of sensor topology and actuator networks of wireless access allowed to reveal advantages, lacks and means of improvement. This led to the development of the "SNOW" simulator (Sensor Network Over Wireless). The following requirements have been identified as the main ones that will provide the necessary environment for conducting experiments on the construction and study of sensor networks:

- support of inhomogeneous network structure;
- the possibility of independent description and simultaneous use in experiments of different components of the network model;
- the maximum approximation of the node behavior description and the protocol to the description of them in the node software;
- the ability to describe arbitrary methods of construction and data exchange protocols in a wireless network;
- simplicity of describing the behavior of network nodes;
- ability to expand the simulator by adding new models;
- the ability to change the level of detail for each of the models;
- the ability to remove arbitrary characteristics of the network or individual nodes in real time;
- powerful tools for visualization of results: communication graphs, graphs;
- the ability to save the initial parameters and results of experiments;
- the ability to conduct a series of experiments with different settings;
- no restrictions on the size of the studied network;
- high speed.

The "SNOW" program is a simulator of discrete events in time, designed to study:

- network formation and reconfiguration processes;
- topology control;
- routing methods in the IS;
- distributed algorithms for the operation of nodes and wireless communication protocols related to the channel, network, transport and session layers of OSI;
- algorithms and methods of building IP, as part of a local area network with wireless access.

The simulator program provides:

- graphical shell to adjust the parameters of the experiment;
- save configuration files and host locations to play the experiment;
- experiment results display in text and graphical formats (graphs of characteristics in real time, map of nodes, coverage areas, communication graph).

The simulator's ability to scale experiments is limited only by the hardware characteristics of the PC on which the simulation is performed and the simulation time itself. As for the functional extension, thanks to the modular architecture it is possible to add any new model for a particular component.

3.1 Simulator architecture

In the article [18] a general description of the simulator architecture is given. There is briefly described and explained the relationships between the components of the network model that are implemented in the developed simulator and the functionality of the simulator kernel.

[18] provides a general description of the simulator architecture, briefly describes the relationships between the components of the network model implemented in the developed simulator, explains the functionality of the simulator core. This article describes the structure of classes in the simulator in more detail and explains the chosen decomposition. In the diagrams mentioned in Figure3 and Figure 4 the main program classes and subclasses of the developed simulator are shown in accordance with:

1. implementation of different types of IP devices for the wireless sensor networks ("Specification of Control Types")
2. implementation of their behavior, ie the exchange of messages in the wireless sensor networks ("Specification of Stages of Work").

The architecture constructed in this way allows to obtain the necessary flexibility in the description of all components of the model. It is also possible to study arbitrary methods of topology control for the network (see “Model of Construction Method” in Figure 4) with simultaneous support of inhomogeneous devices (see “Model of Network Node” in Figure 3).

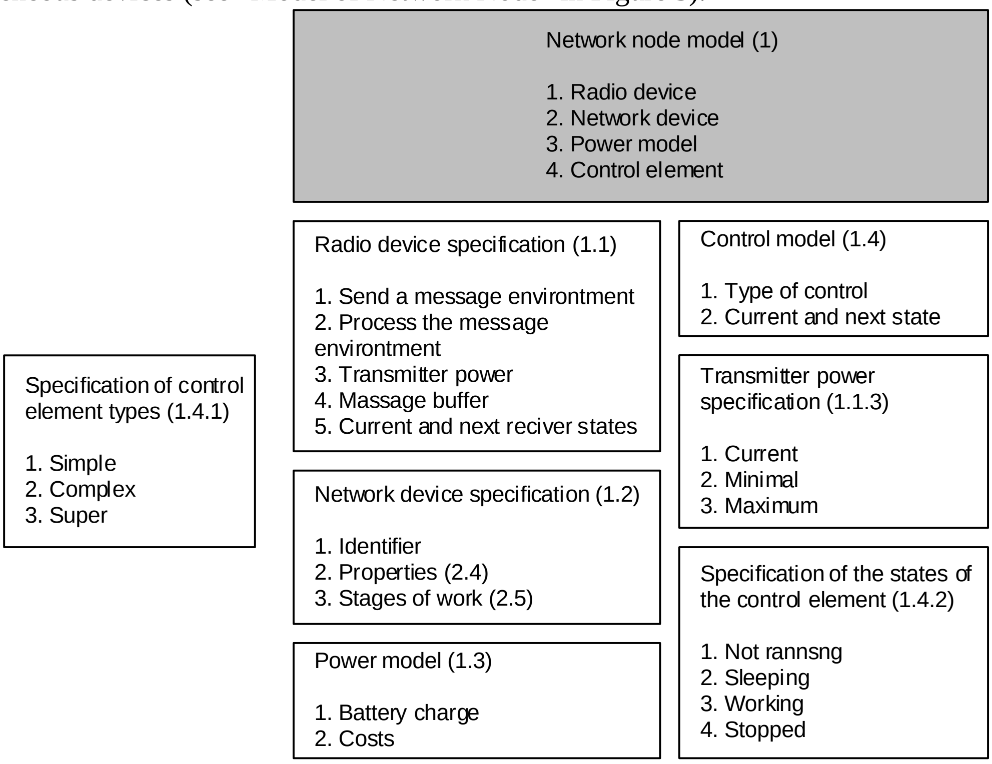

Figure 3: Control element – the network node model and its components.

“Network Node Model”, in particular, can be described arbitrary un combinations of the following components:

- Radio Device – sending and receiving data from the air, allows to adjust the power of the transmitter and the sensitivity of the receiver and to control its activation;
- Network Device – network identification; storing, updating and accumulating information about neighbors, routes, etc;
- Power model – control of battery charge, data transmission and reception costs;

- Control Element – implements the machine states of the node operation and the transition to them in accordance with the current method of the nod work; the basic set includes four states of operation of the node; Each of the components described above can be supplemented with new properties and allows to make changes to existing ones [19, 20].

For the “Topology Control Model” we can describe arbitrary:

- types and structure of messages;
- protocol states and functionality of each of the states;
- incoming message handlers;
- communication radius assignment function.

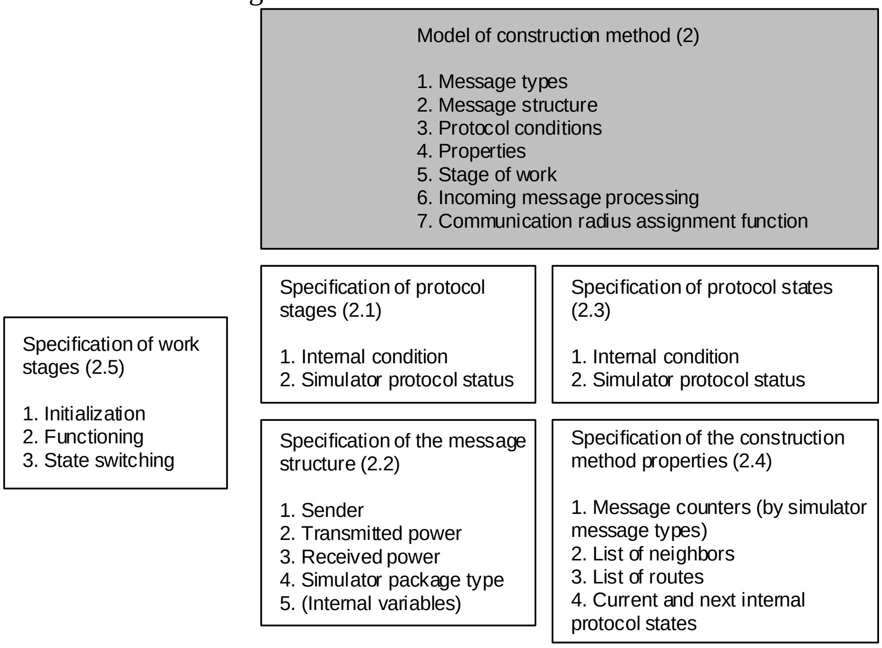

Figure 4: Behavior description – a model of the construction method and its components.

Figure 5 shows the relationships between the main classes of the program. The figure does not show the graphics subsystem.

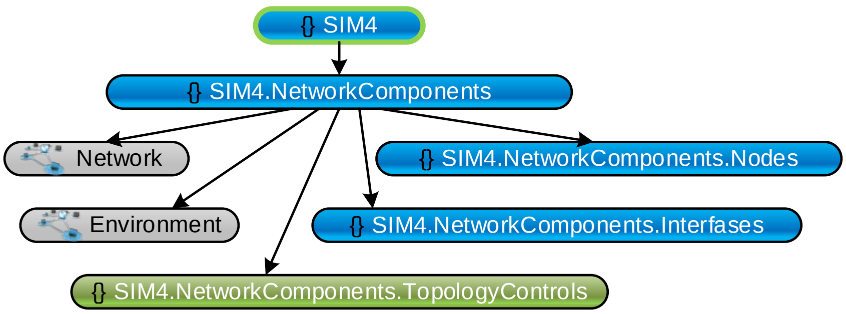

Figure 5: The main classes of the simulator.

Figure 6 – Figure 8 shows the interfaces described for the components of the network model as part of the simulator.

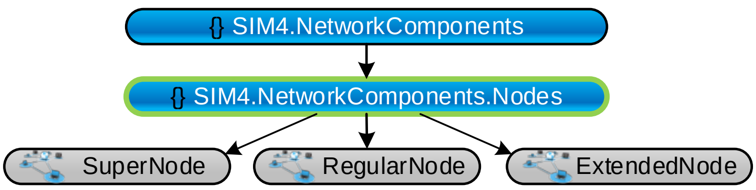

Figure 6: Types of nodes.

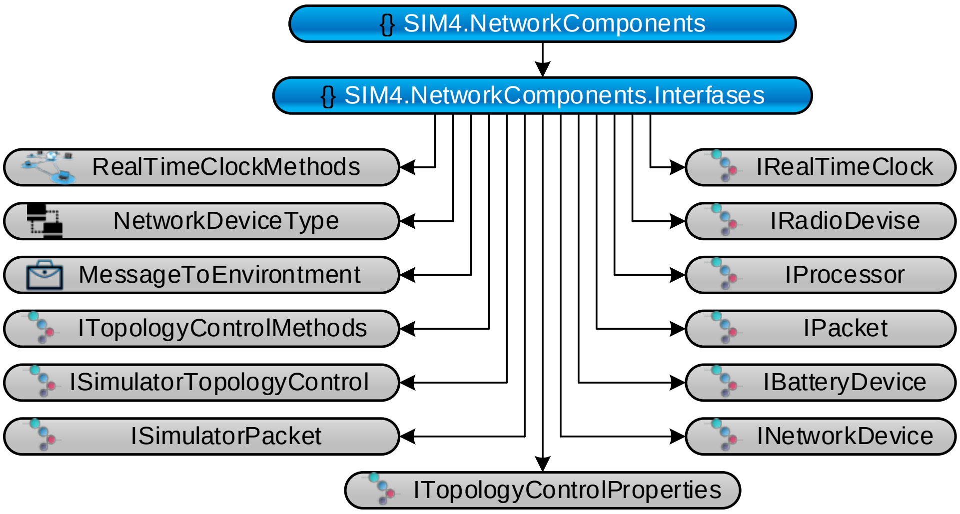

Figure 7: Simulator interfaces.

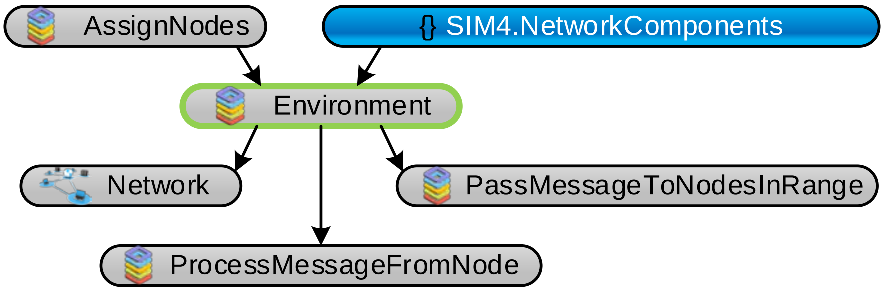

Figure 8: Environment model and its connections.

3.2 Simulator graphical interface

The Table 1 shows an example of setting parameters for sensor networks simulation.

Table 1
Parameters for sensor networks simulation

Parameter	Meaning
Number of nodes	100
Number of monitors	1
Minimum radius, maximum radius, step	4, 30, 2
Launch devices	Simultaneous
Target logical connectivity	3
Initial energy, transmission costs, reception costs per cycle	30000, 2, 3
Number of retransmissions	5
The size of the placement area	100×100
Type of accommodation	Random (same for both experiments)
Device types	"Super", "Simple"

The simulator interface consists of the following windows:

1. Main window (Figure 9)
2. The window for generating the number and type of nodes (Figure 10)
3. Node settings change window (Figure 11)
4. Communication graph view window (Figure 12)

Main window

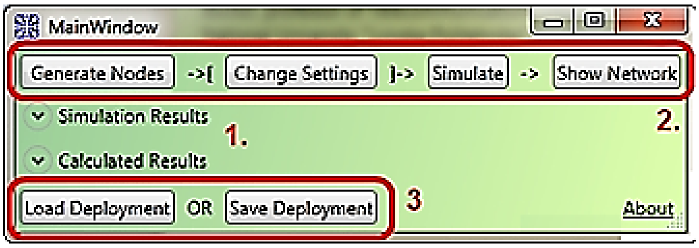

Figure 9: Main window:

1. display elements of calculations and simulation results;
2. the main means of setting up and controlling the simulation (left-right): network generation, experiment setup, start the simulation, display the communication graph;
3. save the current and load the existing configuration of the location.

Node generation window

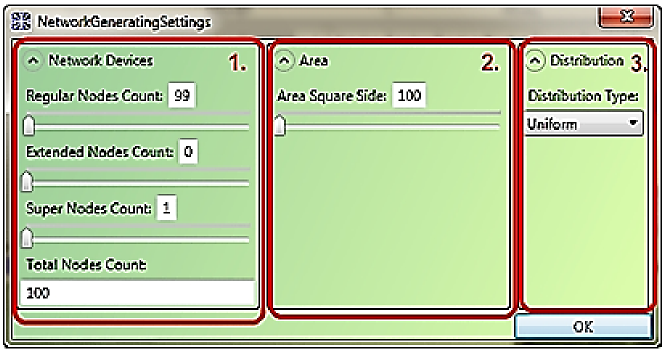

Figure 10: Node generation window:

1. distribution of nodes number by type;
2. the size of the location;
3. distribution of nodes on the plane.

Settings change window

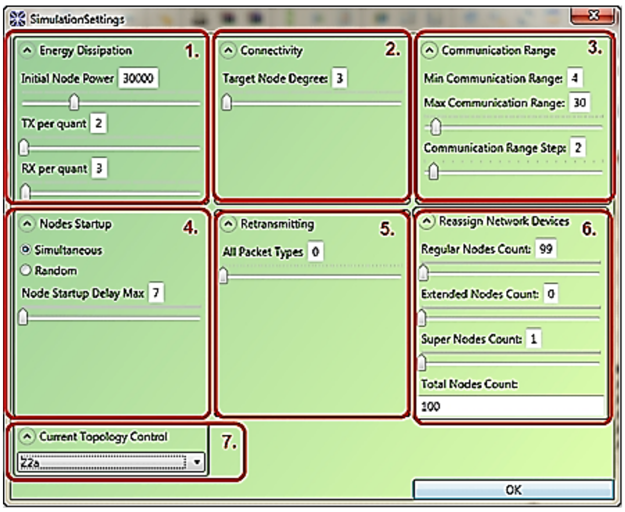

Figure 11: Settings change window:

1. energy consumption parameters;
2. target connectivity;

3. coverage area parameters for the node;
4. type of start of knots;
5. retransmission parameters;
6. change the distribution of nodes by type;
7. choice of construction method.

Simulation window

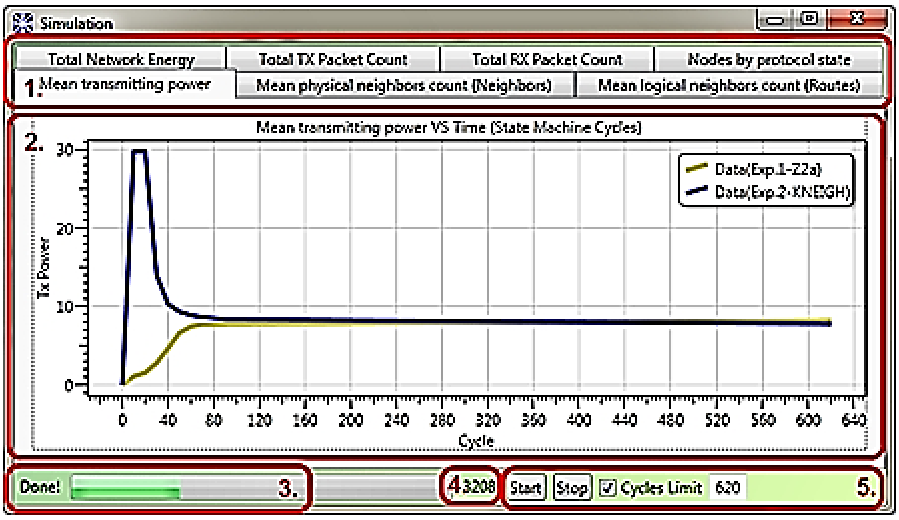

Figure 12: Node generation window:
1. tabs of the removed characteristics;
2. area of real-time graphs characteristics display;
3. the progress of the experiment;
4. time spent on the simulation;
5. simulation control.

Communication graph view window

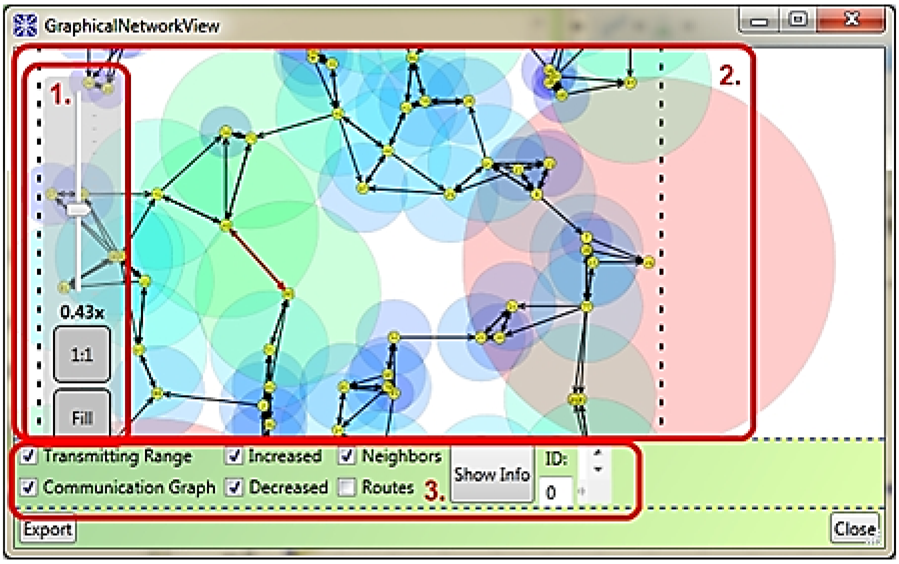

Figure 13: Node generation window:
1. scaling control;
2. display area of communication graphs;
3. display control.

4. Experimental results

The main results of the development and application of the SNOW simulator will be demonstrated by examples of the study of real sensor networks.

Simulation time for the K-NEIGH network topology method of construction and control

Table 2 shows the duration of 100 simulation steps for experiments with different numbers of nodes. The size of the side of the nodes square area was changed to maintain the same density of the location. The side of the square region is calculated by the formula:

$$r = \sqrt{\frac{|N|}{q}} \tag{1}$$

where N is the set of nodes, r is the side of the nodes square area, q is the density of the nodes; the density is equal to 0.01; r is rounded to the nearest larger number that is a multiple of 10.

We believe that devices in the network can be of two types: "Simple Node" (limited autonomous power supply) and "Super Node" (unlimited power).

Instead of assigning the same communication radius to all nodes, we use the function of assigning the communication radius: a gradual increase in the communication radius from the minimum value until we achieve the desired connectivity of the node. For the first method of construction, it is physical, and for the second – logical connectivity of knot.

A number of experiments were performed for networks of different sizes, which aimed to determine the simulator performance on the example of the K-NEIGH topology control method.

The K-NEIGH (K-Neighbors) method involves building a network based on a certain minimum required number for each node neighbors, which ensures the connectivity of the communication graph.

Experiments were performed with off (sequential simulation), partially on and full on parallelization of processes in the simulator core. Table 2 shows the obtained data.

Table 2

Simulation time in different modes of the simulator

Network size	R, units	T, sequential, ms	T, partially parallel, ms	T, parallel, ms
100	100	916	1389	914
200	140	1978	2347	1663
500	220	8343	7647	5073
800	280	20047	16726	11886
1000	320	30350	22228	17243
1200	350	42924	30352	24575
1500	390	67447	44069	39762
1700	410	88466	55174	47393
2000	450	117378	74771	66867
2500	500	186351	110318	98441
3000	550	264805	155691	142234

Sequential simulation here means the use of sequential cycles when performing both state machines of all nodes and when processing (redirecting) messages by the environment. With a partial parallelization, message processing is carried out by the medium and the parallel cycle.

Improving the speed of the simulator – work in parallel mode

One way to increase the productivity is more sparse performance. In the conducted experiments, the removal of all characteristics occurred every 10 steps of the simulation with a total number of steps equal to 200. More frequent removal of characteristics provides more accurate intermediate results and, accordingly, smoothed graphs of the obtained characteristics over time.

Also note, that the speed experiments were performed in the program debugging mode; eliminating the collection of debugging information saves up to 30% of the time. To test this assumption, experiments were repeated for the cases listed in Table 2 color. It was confirmed that when you run the program in normal operation, the gain ranges from 26.6% to 33.8%.

Time costs for experiments in different modes are shown in Figure14.

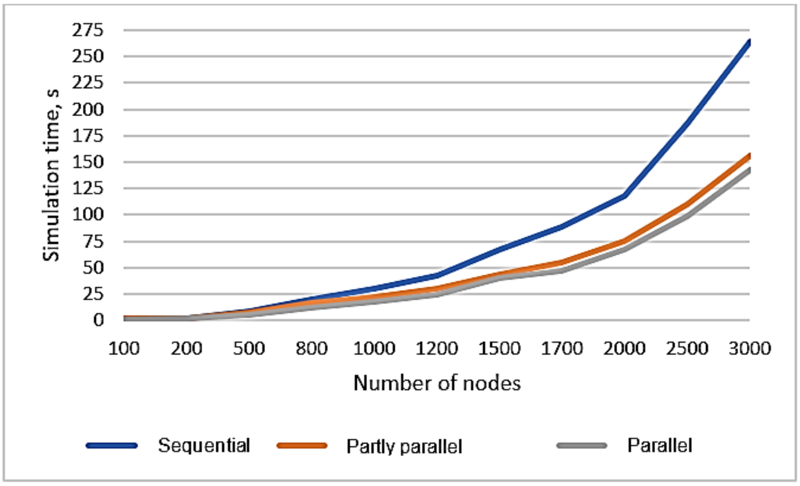

Figure 14: The results of experiments to determine the effect of parallel computing of the simulator performance.

Scaling capabilities and graphical interface of simulation results

As you can see from the Figure14, the use of parallel calculations in the simulator allows you to significantly reduce the time spent on the simulation, while achieving the same results.

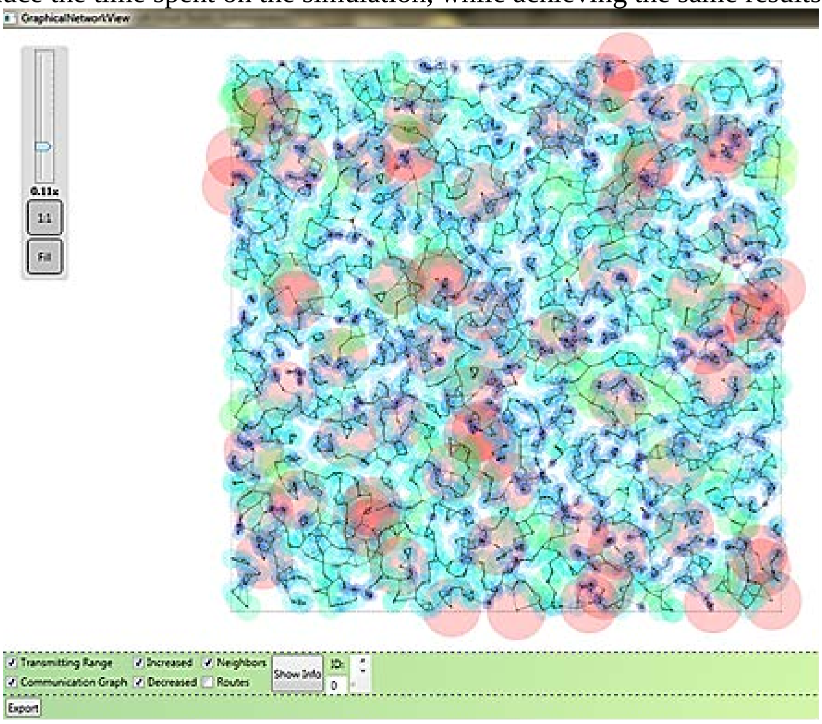

Figure 15: Communication graph after the simulation of the K-NEIGH method in a 3000 elements network.

This effect increases as the size of the network increases. N the Fig. 15, there is given an example of a communication graph for a 3000 elements network by the K-NEIGH construction method, which is obtained by means of the "SNOW" simulator.

5. Conclusion

The wireless network simulator building method adapted for the research on the process of building, forming and reconfiguring the network, topology control, routing methods and maintaining wireless network connectivity, is considered. The simulator allows to explore the distributed algorithms of the node operation and wireless sensor network protocols related to the channel, network, transport and session levels of OSI.

The simulator implements all the requirements (which are determined from the analysis of the advantages and disadvantages of existing simulators and described in section 3), which provides the necessary environment for experiments to build and study sensor networks.

The structure of the simulator program in which the network model is implemented is described in details. Components, classes, interface, environment model and simulator implementation are described. Thanks to the modular architecture, it is possible to use any new model for a particular component. There is a graphical shell for the study parameters adjustment and text displaying in the textual and graphical form (graphs of characteristics in real time, location map of nodes, coverage areas, communication graph).

Examples of work with the simulator are given. One of the simplest ways to build a wireless network is taken as an example. Methods of optimization of the described method of network construction are proposed.

6. Acknowledgements

The authors are appreciative to colleagues for their support and appropriate suggestions, which allowed them to improve the materials of the article.

7. References

[1] M. Zhou, Y. Li, X. Huang, Q. Pu and H. Yuan, Indoor WLAN Intrusion Detection Using Intra-class Transfer Learning with Low Effort, in: Proceedings of the 2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC, Istanbul, Turkey, 2019, pp. 1-6. doi: 10.1109/PIMRC.2019.8904445.

[2] R. Hartung, U. Kulau and L. Wolf, Demo: PotatoScope – Scalable and Dependable Distributed Energy Measurement for WSNs, in: Proceedings of the 2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking, SECON, London, 2016, pp. 1-3. doi: 10.1109/SAHCN.2016.7732981.

[3] C. Schmitt, A. Freitag and G. Carle, CoMaDa: An adaptive framework with graphical support for Configuration, Management, and Data handling tasks for wireless sensor networks, in: Proceedings of the 9th International Conference on Network and Service Management, CNSM' 2013, Zurich, 2013, pp. 211-218. doi: 10.1109/CNSM.2013.6727839.

[4] S. H. Gade and S. Deb, HyWin: Hybrid Wireless NoC with Sandboxed Sub-Networks for CPU/GPU Architectures, IEEE Transactions on Computers 66 7 (2017) 1145-1158. doi: 10.1109/TC.2016.2643668.

[5] C. Peng, K. Qian and C. Wang, Design and Application of a VOC-Monitoring System Based on a ZigBee Wireless Sensor Network, IEEE Sensors Journal 15 4 (2015) 2255-2268. doi: 10.1109/JSEN.2014.2374156.

[6] M. R. M. Kassim and A. N. Harun, "Using Wireless Sensor Network to determine pollination readiness of oil palm flower, in: Proceedings of the 2015 9th International Conference on Sensing Technology, ICST, Auckland, 2015, pp. 59-64. doi: 10.1109/ICSensT.2015.7438365.

[7] B. Durnyak, B. Havrysh, O. Tymchenko, M. Zelyanovsky, O. O. Tymchenko and O. Khamula, Intelligent System for Sensor Wireless Network Access: Modeling Methods of Network Construction, in: Proceedings of the 2018 IEEE 4th International Symposium on Wireless Systems within the International Conferences on Intelligent Data Acquisition and Advanced

Computing Systems, IDAACS, Lviv, 2018, pp. 93-97. doi: 10.1109/IDAACS-SWS.2018.8525792.
[8] M. Logoyda, M. Nazarkevych, Y. Voznyi, S. Dmytruk, O. Smotr, Identification of Biometric Images using Latent Elements. CEUR WS 2488 (2019) 99-108.
[9] M. M. R. Mozumdar, A. Ganesan and A. Ameri, Synthesizing Sensor Networks Backbone Architecture for Smart Buildings, IEEE Sensors Journal 14 12 (2014) 4273-4283. doi: 10.1109/JSEN.2014.2346654.
[10] M. J. Saikia, G. Cay, J. V. Gyllinsky and K. Mankodiya, "A Configurable Wireless Optical Brain Monitor Based on Internet-of-Things Services, in: Proceedings of the 2018 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), Msyuru, India, 2018, pp. 42-48. doi: 10.1109/ICEECCOT43722.2018.9001456.
[11] M. Kemal, F. Iov, R. Olsen, T. Le Fevre and C. Apostolopoulos, On-line configuration of network emulator for intelligent energy system testbed applications, in: Proceedings of the AFRICON 2015, Addis Ababa, 2015, pp. 1-4. doi: 10.1109/AFRCON.2015.7331979.
[12] H. Saidi, M. Turki, Z. Marrakchi, M. S. Ben Saleh and M. Abid, Embedded FPGA accelerator for Wireless Sensor Network nodes, in: Proceeedings of the 2016 11th International Design & Test Symposium (IDT), Hammamet, 2016, pp. 313-318, doi: 10.1109/IDT.2016.7843061.
[13] M. B. Duggimpudi, A. Moursy, E. Ali and V. V. Raghavan, An Ontology-Based Architecture for Providing Insights in Wireless Networks Domain, in: Proceedings of the 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI), Omaha, NE, 2016, pp. 473-478. doi: 10.1109/WI.2016.0078.
[14] O. Tymchenko, O. O. Tymchenko, B. Havrysh, O. Khamula, O. Sosnovska and S. Vasiuta, Efficient Calculation Methods of Subtraction Signals Convolution, in: Proceedings of the 2019 IEEE 15th International Conference on the Experience of Designing and Application of CAD Systems (CADSM), Polyana, Ukraine, 2019, pp. 1-4, doi: 10.1109/CADSM.2019.8779250.
[15] A. Kasture, A. Raut and S. Thool, Visualization of Wireless Sensor Network by a Java Framework for Security in Defense Surveillance, in: Proceedings of the 2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies, Nagpur, 2014, pp. 256-261. doi: 10.1109/ICESC.2014.49.
[16] A. Guerrieri, L. Geretti, G. Fortino and A. Abramo, A service-oriented gateway for remote monitoring of building sensor networks, in: Proceedings of the 2013 IEEE 18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Berlin, 2013, pp. 139-143. doi: 10.1109/CAMAD.2013.6708105.
[17] T. Yilmaz and O. B. Akan, Utilizing terahertz band for local and personal area wireless communication systems, in: Proceedings of the 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD’2014, Athens, 2014, pp. 330-334. doi: 10.1109/CAMAD.2014.7033260.
[18] Z. Fan, Z. Bai, X. Zhang, S. Rahardja and J. Chen, AUC Optimization for Deep Learning Based Voice Activity Detection, in: Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP’2019, Brighton, United Kingdom, 2019, pp. 6760-6764. doi: 10.1109/ICASSP.2019.8682803.
[19] F. Wu, C. W. Tan, M. Sarvi, C. Rudiger and M. R. Yuce, Design and Implementation of a Low-Power Wireless Sensor Network Platform Based on XBee, in: Proceedings of the 2017 IEEE 85th Vehicular Technology Conference (VTC Spring), Sydney, NSW, Australia, 2017, pp. 1-5. doi: 10.1109/VTCSpring.2017.8108667.
[20] T. Nishio, M. Morikura and K. Yamamoto, Heterogeneous media communications for future wireless local area networks, in: Proceedings of the 2015 IEEE International Conference on Consumer Electronics, ICCE, Las Vegas, NV, 2015, pp. 637-640. doi: 10.1109/ICCE.2015.7066560.