**DOI: https://doi.org/ 10.15276/aait.04.2019.4**
**UDC 004.738:004.94**

# BEHAVIORAL VERIFICATION OF INTERNET OF THINGS SYSTEMS BY PETRI NETS

**Oleksandr N. Martynyuk**[1)]
ORCID: http://orcid.org/0000-0003-2366-1920, anmartynyuk@ukr.net, Scopus ID: 57103391900
**Oleksandr V. Drozd**[1)]
ORCID: http://orcid.org/0000-0001-8305-2217, drozd@ukr.net, Scopus ID: 55388226700
**Sergey A. Nesterenko**[1)]
ORCID: http://orcid.org/0000-0002-3053-0374, sa_nesterenko@ukr.net, Scopus ID: 55386373800
**Tamem Ahmesh**[1)]
ORCID: http://orcid.org/0000-0002-0482-2656, tamim.nor@yahoo.com
[1)] Odessa National Polytechnic University, 1, Shevchenko Avenue. Odesa, 65044, Ukraine

## ABSTRACT

The rapid development, implementation in all spheres of human activity and the growing responsibility of the functions of the Internet of things systems tighten and complicate the requirements for the reliability of their design decisions at the development stages and operability during the implementation of implementations. Well-known methods of verification of projects and implementations are based on the means of systemic, structural, functional, design and technological analysis and synthesis of Internet of things systems. However, their capabilities do not underestimate the feasibility of developing formalized models and verification methods. This study presents the elements of technology and the steps of the behavioral verification methodology of functional level projects for Internet of things systems represented using Petri nets. General verification is represented by three stages - the analysis of the correctness of the general structural and functional properties, the actual verification of inter-level and inter-component interactions, behavioral online and offline testing in the class of functional type errors. In the proposed analysis, the basic entities and relationships of the Internet of things systems are determined and verified architectural level, defining the structure, components, functions, interfaces, asynchronous-event interactions and represent elements of Petri nets - their positions, transitions, arcs, functions, markup. Online and offline testing for dynamic verification of the behavior of the Internet of things systems is carried out on the basis of, respectively, the background or special formation of many process threads in the Petri net, activated during its modeling and covering the Petri net objects. This paper presents general estimates of resource and time costs for the design of Internet of things systems without verification and with verification, showing their reduction in the event of design errors, redesign and application of verification. Verification is illustrated by the example of Petri nets simulating an automatic lighting system.

**Keywords**: Internet of Things Systems; Behavioral Verification; Petri Net; Coverage of Verified Properties; Complexity of Verification

***For citation*: Oleksandr N. Martynyuk, Oleksandr V. Drozd, Sergey A. Nesterenko, Tamem Ahmesh. Behavioral Verification of Internet of Things Systems by Petri Nets. *Applied Aspects of Information Technology*. 2019; Vol.2 No.4: 295–303. DOI: https://doi.org/10.15276/aait.04.2019.4.**

## INTRODUCTION

The rapid development of the Internet has led to the emergence of a new concept for the Internet of Things (IoT). A feature of IoT is that the union of computers and people is transformed into a union of intellectual things [1, 2], [3]. Currently, a developed concept and related technologies have been formed in IoT, and here they are being rapidly deepened and expanded with the prospect of covering the IoT of the entire global network. It can be noted that the Internet is transforming into a significantly sophisticated, intelligent and truly global version of IoT. This is due, in particular, to the fact that the number of devices that provide and use an expanding list of Internet services is growing exponentially, now reaching almost everyone with the help of new communication technologies. A powerful distributed and shared source of a wide variety of information, computing, communication, management, intelligent services provided to end users has appeared and is rapidly developing.

The conceptual borrowed multi-agent IoT, involving the use and development of the properties of autonomy, mobility, intelligence and cooperativeness, provides a high level of flexibility, operational reconfigurability of IoT to the current state of the hosting environment [4, 5], [6, 7].

IoT is an approach to connecting a wide variety of services received from various sources on any virtual platform or Internet infrastructure. In 1999, Bill Joy defined the connections between devices in the Internet taxonomy [8, 9], and Kevin Ashton coined the term “Internet of Things” for related devices. The basic idea of IoT is to provide the possibility of an autonomous exchange of services between uniquely identifiable devices in the real world, which are now increasingly supporting radio frequency identification (RFID) and wireless sensor networks (WSNs), which allow them to make independent decisions depending on what action performed [7].

Many of the indicated integrable properties and mechanisms of IoT, both borrowed and proprietary, the responsibility, and often the criticality of the application, place high demands on the reliability, design and functioning of IoT. In particular, verification of projects and realizations of IoT implementations at system-functional level seems relevant and not trivial [9, 10], [11, 12].

Applied mathematical base of verification of systems of Internet of Things (IoT) and their projects on high functional level may be based on its formal specifications with the use of corresponding high-level structural and functional linguistic foundation. This base may include automata [10] and Petri nets [13, 14], [15, 16], [17, 18].

## THE PURPOSE OF THIS ARTICLE

The main goal of this work is increase of completeness and accuracy of modelling and verification. Tasks to be solved include behavioral verification and testing of architectures and processes for IoT systems on the basis of Petri nets.

## SYSTEM BEHAVIORAL VERIFICATION OF IOT SYSTEMS

The objects of behavior of IoT are taken as input for Petri Nets and include: a) specifications of the technical description of the architecture of components, subsystems IoT and IoE-based systems, as well as such systems, defining the structure of topological relationships, functions, information objects, interfaces of topological interactions, the temporal behavior of functions and scenarios; b) previously prepared Petri nets, set accordingly behavioral models of process, components, IoT systems, for which system behavior verification are needed [19, 20], [21, 22].

The following objects are considered as output objects for Petri Nets: the obtained correct, verified Petri nets, representing functional models of process, components, subsystems of IoT systems, system verification, obtained special conditions, parameters and scenarios of such verification, and also special the results of the fulfillment of conditions, the application of parameters and scenarios [23, 24], [25, 26], [27].

Models and methods for analyzing the functioning of Petri Netscan are used in processes of verification of the various aspects of behavior of the IoT systems and their components [28, 29].

Features in verification of processes in functioning of the IoT systems, their components on the basis of their representation by asynchronous, multiprocessing enhanced and hierarchical Petri nets.

Simulation is the imitation of the operation of a real-world process or system over time. The act of simulating something first requires that a model be developed, this model represents the key characteristics or behaviors/functions of the selected physical or abstract system or process. The model represents the system itself, whereas the simulation represents the operation of the system over time. As before, verification are independent procedures that are used together for checking that service or system meets requirements and that it fulfills its intended purpose. The behavioral online and offline testing the conformity of the behavior of the system under check to the behavior of the reference system, in the mode, respectively, for the first, basic operating functioning and, for the second, specific testing functioning.

*Behavior modeling of features for functioning of IoT systems using Petri Nets.* Behavior modeling is performed taking into scenarios and functions of the architecture of IoT for interactions of ports and interfaces into inter-component structure. The functional features of IoT and also their processes affect the classification of Petri Nets with special functions input/output, storage, processing for IoT.

Behavioral analysis of IoT systems is focused on the modeling and verification of the basic, component, interface and subsystem functions presented at the system level of the IoT architecture.

Thus, these tasks are defined as follows actions correctness, verification, testing [28].

Correctness proves existence of reference structural properties and includes the two stages.

– First stage and its steps for correctness confirms, that the model has the preference properties of: absence of static locks; completeness; unambiguous correspondence of states; lack of redundancy; limitedness; lack of dynamic locks; self-synchronization; partial correctness; complete correctness; security; liveliness.

– Second stage and its steps lowers the dimension of the model of achievable states due steps: analysis of the achievable states and markings; structural and functional decomposition; previously created "equivalent" states; limiting the number of parameters and detectable errors.

Verification proves compatibility of the specifications for the service objects of verifiable and adjacent levels and includes the two stages.

– First stage proves that specification of the lower level, that are used by these service objects, is consistent with the description provided by the verified level.

– Second stage proves that specification of the higher level, that uses these service objects, is consistent with the description provided by the verified level.

Behavioral online and offline testing proves existence of special subject functional properties and includes four stages.

– First stage consists in verifying, that the behavior of the system on the conceptual boundary with the environment corresponds to the intended one.

– Second stage allows to get test scenarios, recognizing and checking experiments in terms of abstract service primitives and data elements of the system.

– Third stage consists in passive recognizing experiment by behavioral on-line testing. This experiment is based on a method of recognizing behavioral automata check in the external flow of the system's operational functioning based on the identification of reference states in the first step. Experiment establishes the correspondence of the reference and verified models by searching for recognizing fragments in a fixed operational behavior of the real system in the second step.

– Four stage executes an active checking experiment by behavioral offline testing. This experiment is based on a formal method of constructing behavioral automata checks in the internal specially formed flow of test functioning of the system based on the identification of reference states in the first step. Experiment establishes the correspondence of the reference and verified models by the embedding of checking fragments in the constructed test behavior of the real system in the second step.

## REALIZING OF BEHAVIORAL VERIFICATION OF IOT SYSTEMS

The aim of the work is applying the methods of behavior verifying of the functional models of IoT systems by Petri nets environment

General objectives include familiarization with: the fundamentals of model verification, in particular, based on the use of intuitive – “manual”, full-selection, and automata testing in the “promotion” method; the principles of operation and the basics of using the environment CPN Tools, which provides a formal performs the basic functional, event-time modeling and verification of IoT systems and their projects.

Tasks of realizing of verification includes: constructing of models of IoT system implemented at the application level; performing verification of IoT system models based on CPN Tools environment.

Goal of realizing of verification includes: analysis and subject optimization of the complexity of the design, verification and redesign of the IoT systems; assessment and subject optimization of the length, multiplicity and completeness of verification checks for designing of IoT systems.

Procedure for performing formal Petri nets models of the architecture of some IoT system and its verification scenarios include eleven realizing stage:

1. Construction behavioral models of the architecture selected according to the IoT system task – activity diagram and state diagram.

2. The upper estimate of the complexity of the behavioral analysis of input UML diagram is performed using a simplified formula for basic entities and relations.

$$C_{UML} = a*(n_e+n_e^2+3*n_r), \tag{1}$$

where: $n_e$ is the number of classifiers of the diagram entities;

$n_e^2$ is the number of cells in a square matrix of possible relationships between entities;

$n_r$ is the number of real assigned ratios of the diagram, the multiplier “3” means the need to consider both the relation r itself and the two entities e1 and e2 incident to it; a – conditional coefficient of abstraction level (for technical specifications – 3, structural-functional – 2, object-component – 2, event-time – 1, automata-algorithmic – 1).

3. Construction a script for behavioral verification of properties based on a full-choice testing method for constructing a path covering all entities for each of the selected diagrams and determining the path length.

4. The upper estimate of the computational complexity of the covering path (including its construction) using the simplified formula:

$$C_{Path} = a*(n_e^2/2+n_r). \tag{2}$$

5. The upper estimate of the reduced complexity of the construction due to the use of behavioral verification script in comparison with the case of redesign in case of an error, when the complexity of the analysis and construction of input UML diagram is doubled, using the simplified formula:

$$\begin{gathered} C\Delta_C = 2*C_{UML} - (C_{UML}+C_{Path}) = \\ =a*(n_e+ n_e^2/2 +2*n_r). \end{gathered} \tag{3}$$

6. Visual construction of Petri nets in the CPN Tools according to the input UML diagrams of the IoT system based of activities and states.

7. Determination of the upper estimate of the complexity of the analysis and the construction of Petri nets using a simplified formula for their entities – positions, transitions, chips and relations for them:

$$C_{Petri} = a*(n_e+n_q+4*n_r), \tag{4}$$

where: $n_{ep}$ is the number of position classifiers, $n_{et}$ is the number of transition classifiers,

$n_{em}$ is the number of chip token classifiers, $n_e$=$n_{ep}$+$n_{et}$+$n_{em}$; $n_q$=$n_{ep}$*$n_{et}$*$n_{em}$ is the number of cells in the three-dimensional matrix of possible relationships between entities; $n_r$ is the number of real assigned ratios of the diagram, the multiplier "4" means that it is necessary to consider both the relation r itself and the three incident entities $e_{1p}$, $e_{2t}$, $e_{3m}$; a – conditional coefficient of abstraction level (for technical specifications – 3, structural-functional – 2, object-component – 2, event-time – 1, automaton-algorithmic - 1).

8. Constructing of scenarios of static and dynamic verification of selected properties of Petri net based on CPN Tools for:
– step-by-step simulation;
– end-to-end modeling;
– constructing a graph of attainable markings;
– matrix definition of position and transition invariants.

9. Construction of the test, as a covering all the positions and transitions of the Petri net based on recurring behavioral testing, and determining the length of this path.

10. The upper estimate of the complexity of the covering path using the simplified formula:

$$C_{PetriPath} = a*(n_q/3+n_r). \tag{5}$$

The upper estimate of the reduced complexity of the construction due to the use of the behavioral verification scenario in comparison with the case of re-design in case of an error when the complexity of analyzing and building the Petri net doubles, using the simplified formula:

$$C\Delta_C = 2*C_{Petri} – (C_{Petri}+C_{PetriPath}) = \\ =a*(n_e+2n_q/3+3*n_r). \tag{6}$$

*Examples of verification of Petri nets*

The design and verification of the architecture of the IoT system selected according to the setting option using Petri nets. Two type of Petri nets demonstrate the design and verification of IoT system.

Activity Chart. The first step of the behavioral description of IoT system is the actions that can be performed, a convenient type of diagrams for this is an activity diagram, and they can also use a simpler and more familiar language of flowcharts — graph diagrams – instead. They describe the main user actions when working with the system. The first thing that is required of the user is to enter the premises, thereby causing the motion sensor tripped to work, then enter the flow time of the day. Next, the system includes the corresponding mode. After that, the user sets a valid value for the light level for the specified mode. Further, the user is given a choice: to continue working with the system or not. In case of agreement, the system returns to the time setting.

The logic of work is in the form of activity diagrams (see Fig. 1)

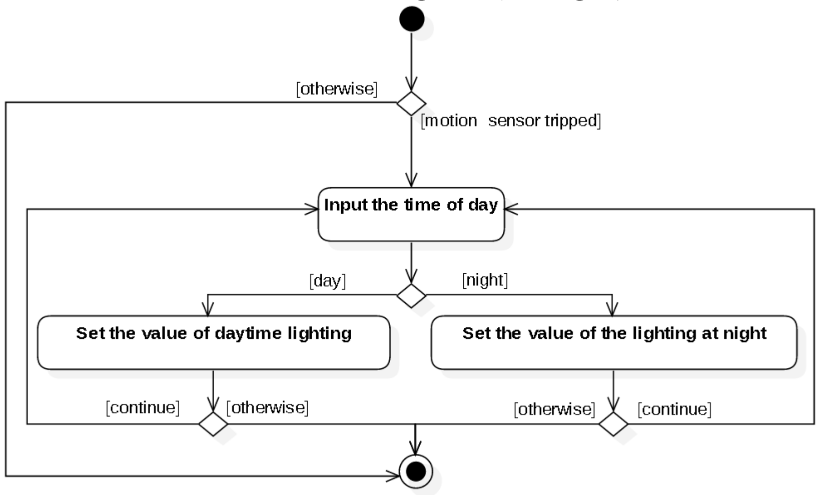

*Fig. 1.* **Activity diagram of the lighting system**
***Source*: compiled by the author**

Entity verification for the diagram:
– Actions: enter the stream time of day (Input the time of day), set the allowable value of the lighting level for day mode, and set the allowable value of the level of light for night mode.
– Relations-transitions: start → branching; branching → time; time → branching; branching → values by day; values by day → branching; branching → time; branching → end; branching → values at night; values at night → branching; branching → time; branching → end; branching → end.
– Constructions for registration transitions: branching; compound.

The verification of entities of a diagram of the illumination system activity is performed on the basis of a test of coverage – an interactive program enumeration of complete simple paths from start to final vertex, covering all verification entities.

Set of complete simple paths includes six paths:

– (start, branch, time set, branch, set by day, branch, and end) is a simple way;

– (start, branch, set time, branch, set values at night, branch, end) is a simple way;

– (start, branch, set time, branch, set values by day, branch, (set time, branch, set values by day, branch) end) - a simple loop is nested;

– (start, branch, set time, branch, set values by day, branch, (set time, branch, set values at night, branch) end) – a simple loop is nested;

– (start, branch, set time, branch, set values at night, branch, (set time, branch, set values at night, branch) end) – a simple loop is nested;

– (start, branch, set time, branch, set values at night, branch, (set time, branch, set values by day, branch) end) – a simple loop is nested.

Consequently, the combinatorial-enumerated number of possible paths that differ in the input data is six. The upper complexity estimates are of the form:

$$C_{UML} = a*(n_e+n_e^2+3*n_r) =$$
$$=3*(7+7^2+3*6) = 222$$
$$C_{Path} = a*(n_e^2/2+n_r) =$$
$$=3*(7^2/2 +6) = 91,5.$$
$$L_{Path} = 7+9+6 = 22$$
$$\Delta_C = a*(n_e+ n_e^2/2 +2*n_r) =$$
$$=3*(7+7^2/2 +2*6) = 130,5.$$

Reducing the computational complexity of analysis and design through the use of an automated local verification script amounted to 130.5 conventional units of analysis.

State diagrams. State diagrams (statechart diagrams) show the possible states of components or the system as a whole, transitions between them in response to any events and actions performed during this. States can be arranged hierarchically, they can be decomposed into parallel substrates. A subsystem can have global (within its framework) variables storing some data. The values of these variables are common parts of all depicted states. The peculiarity of the state diagram is the use of a modified association of state transitions, not transitions — message exchanges, as well as the possibility of representation of all scenarios as relations. And also in the temporal, rather than spatial, character of the representation of transition relations.

Verification of the state diagram, checking all the scenarios or their essential part, is more complex (up to NP-complex for non-trivial diagrams with tens and hundreds of entities). Reducing the NP-complexity can be achieved through the use of behavioral testing and decomposition methods.

The logic of work is in the form of state diagrams (see Fig. 2).

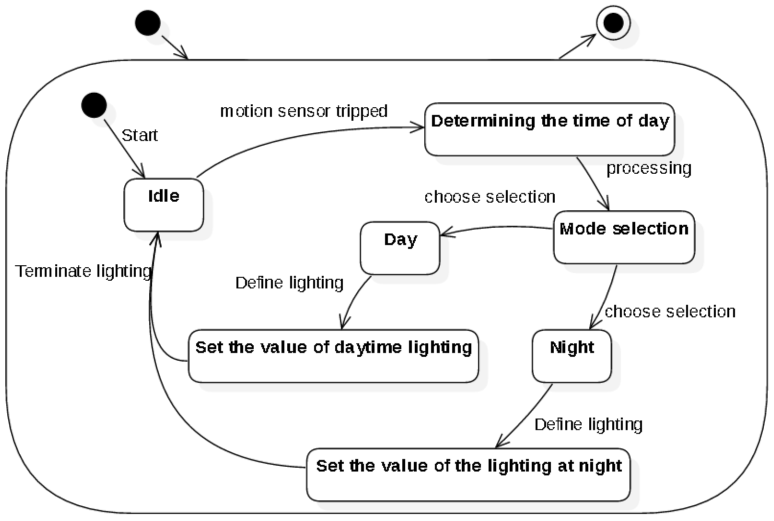

*Fig. 2.* **Lighting system statechart diagram**
***Source*: compiled by the author**

Entities verification of the state diagram:

– States: Idle, Determining the time of day, Mode selection, Day, Night, Set the value of daytime lighting, Set the value of the lighting at night, StartOut, End, StartIn.

– Relations of the modified association - state transitions: StartOut →start()→system; system →start()→StartIn; StartIn →start()→Idle; Idle → motion sensor tripped → Determining the time of day; Determining the time of day → processing → Mode selection; Mode selection → choose selection → Day; Day → Define lighting → Set the value of daytime lighting; Set the value of daytime lighting → Terminate lighting → Idle; Mode selection → choose selection → Night; Night → Define lighting → Set the value of the lighting at night; Set the value of the lighting at night → Terminate lighting → Idle; StartPage →back()→system; system →back()→ End.

A formal verification of entities of a given, relatively simple state diagram is performed on the basis of a complete transition coverage test — one of the possible Eulerian paths through association relations (transitions – calls to the corresponding methods) from the initial Start entity to the final End, covering all verification entities, with registration of results and their interactive verification, in particular, of the property values against the standards (architectural objects and association relations).

The upper complexity estimates are the form:

$$C_{UML} = a*(n_e+n_e^2+3*n_r) =$$
$$=3*(10+10^2+3*11) = 429$$
$$C_{Path} = a*(n_e^2/2+n_r) =$$
$$=3*(10^2/2 +11) = 183$$
$$L_{Path} = 1+2+9 = 12$$
$$\Delta_C = a*(n_e+ n_e^2/2 +2*n_r) =$$
$$=3*(10+10^2/2 +2*11) = 246.$$

Reducing the complexity of analysis and design through the use of behavioral verification script amounted to 246 conventional units of analysis.

## CONCLUSIONS

The formation of a knowledge system of a formalized analysis and synthesis of Internet of Things systems is becoming an important part of the process of training specialists in the field of computer science. Such formalization presupposes a formal study of the models of the components of IoT as a whole, the verification of their properties and the process of functioning. At the system behavioral level of IoT, research is carried out for components, subsystems and IoT as a whole, taking into account their functional features.

In this work behavioral modeling and simulation of IoT systems is considered using Petri nets. Behavioral models, correctness analysis, verification
and testing of architectures and processes of IoT systems based on Petri nets, that are focused on the analyses of general, component and interface functions, presented at the behavior of the IoT.

## REFERENCES

1. Pallavi, Sethi & Smruti, R. "Sarangi Internet of Things: Architectures, Protocols, and Applications". *Journal of Electrical and Computer Engineering.* Article ID 9324035. 2017; Vol. 5: 25. DOI: https://doi.org/10.1155/2017/9324035.

2. Kharchenko, Vyacheslav, Ah Lian Kor, Rucinski, Andrzej (Eds). "Dependable IoT for Human and Industry: Modeling, Architecting, Implementation"/ *River Publishers Series in Information Science and Technology*. 2018. 450 p.

3. Tara, Salman "Networking Protocols and Standards for Internet of Things". https://www.cse.wustl.edu/~jain/cse570-15/ftp/iot_prot/index.html, Internet source.

4. Talari, Saber, Shafie-khah, Miadreza, Siano, Pierluigi, Loia, Vincenzo, Tommasetti, Aurelio & Catalão, João P. S. "A Review of Smart Cities Based on the Internet of Things Concept". *Publ. Energies.* 2017; 10, 421: 23 p. DOI: http://www.mdpi.com/1996-1073/10/4/421/pdf doi.org/10.3390/en10040421.

5. Strielkina, A. A., Uzun, D. D., Kharchenko, V. S. & Tetskyi, A. H. "Modelling and Availability Assessment of Mobile Healthcare IoT Using Tree Analysis and Queueing Theory". *In book: "Dependable IoT for Human and Industry: Modeling, Architecting and Implementation*". Kharchenko, Vyacheslav, Ah Lian Kor, Rucinski, Andrzej (Eds.). *Publ. River Publishers Series in Information Science and Technology*. 2018.

6. Dovgal, V. A., Dovgal, D. V. "Management of resources on the Internet of Things", *Distance educational technologies: proceedings of the II Russian scient.- pract. conf.* Yalta: Simferopol: *Publ. ARIAL.* 2017. p. 168–173.

7. Ashton, Kevin. "That "Internet of Things" Thing". *RFID Journal.* URL: http://www.rfidjournal.com/articles/pdf?4986. 22 June 2009.

8. Evans, D. "Internet of Things", *Cisco, white paper*. https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf. Internet source. 2009.

9. Dovgal, V. A. & Dovgal, D. V. "Security Issues and Challenges for the Intellectual Networks Founded on the Internet of Things" (in Russian). *The Bulletin of the Adyghe State University*. Ser. Natural-Mathematical and Technical Sciences. 2017; Iss.4: 140–147. URL: http://vestnik.adygnet.ru.

10. Esparza, Javier. "Automata theory an Algorithmic Approach. Lecture Notes". Available from: https://www7.in.tum.de/~esparza/autoskript.pdf. August 26, 2017. 321 p.

11. Kondratenko, Y., Kozlov, O., Topalov, A., Korobko, O. & Gerasin, O. "Automation of Control Processes in Specialized Pyrolysis Complexes Based on Industrial Internet of Things". *In book: "Dependable IoT for Human and Industry: Modeling, Architecting, Implementation"*. Kharchenko, Vyacheslav, Ah Lian Kor, Rucinski, Andrzej (Eds.). *River Publishers Series in Information Science and Technology*. 2018.

12. Boyarchuk, A., Kharchenko, V., Illiashenko, O., Maevsky, D., Phillips, C., Plakhteev, A. & Vystorobska, L. "Internet of Things for Human and Industry Applications: ALIOT Based Curriculum*" In book: "Dependable IoT for Human and Industry: Modeling, Architecting, Implementation"*. Kharchenko, Vyacheslav, Ah Lian Kor, Rucinski, Andrzej (Eds.). *River Publishers Series in Information Science and Technology*. 2018.

13. Hahanov, Vladimir, Litvinova, Eugenia & Chumachenko, Svetlana. "Cyber Physical Computing for IoT-driven Services". *Publ. Springer*. 2017. 279 p.

14. Desel, Jorg, Esparza & Javier Free "Choice Petri Nets". *Cambridge University. Press, Cambridge*. Available from: https://www7.in.tum.de/~esparza/fcbook-middle.pdf. 1995. 256 p.

15. Kleijn, J., Yakovlev, A. (Eds). "Petri nets and Other Models of Concurrency". *ICATPN. Lecture Notes in Computer Science*. ISBN 978-3-54073093-4. *Publ. Springer-Verlag*. 2007; Vol: 4546515 p.

16. Poliakov, I., Mokhov, A., Rafiev, A., Sokolov D. & Yakovlev, A. "Automated Verification of Asynchronous Circuits Using Circuit Petri Nets". *Proceedings of the 14th IEEE International Symposium on Asynchronous Circuits and Systems,* Newcastle upon Tyne. UK. April 2008. p.161–170. DOI**:** https://doi.org/ 10.1109/ASYNC.2008.18.

17. Zaitsev, D. A. "Toward the Minimal Universal Petri Net". *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. 2013. p. 1–12.

18. Westergaard, Michael. "CPN Tools", Eindhoven: https://westergaard.eu/wp-content/uploads/2010/09/CPN-Tools.pdf. Internet source. 2010. 46 p.

19. Sugak, A., Martynyuk, O. & Drozd, O. "The Hybrid Agent Model of Behavioral Testing", *International Journal of Computing*. Ternopol: Ukraine. 2015; Vol.14 Issue 4: 232–244.

20. Kharchenko, V., Gorbenko, A., Sklyar, V. & Phillips, C. "Green Computing and Communications in Critical Application Domains: Challenges and Solutions". *IX International Conference of Digital Technologies,* Zhilina: Slovak Republic. 2013. p. 191–197.

21. Sugak, A., Martynyuk, O. & Drozd, O. "Models of the Mutation and Immunity in Test Behavioral Evolution". *Proceedings of the 2015 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*. Warsaw: Poland. 2015. p. 790–795. DOI: https://doi.org/ 10.1109/IDAACS.2015.7341411.

22. Martynyuk, O., Sugak, A., Martynyuk, O. & Drozd, O. "Evolutionary Network of Testing of the Distributed Information Systems". *Proceedings of the 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, https://ieeexplore.ieee.org/document/8095215. Bucharest: Romania. 2017. p. 888–893.

23. Wang, Anduo. "Formal Analysis of Network Protocols". *University of Pennsylvania Department of Computer and Information Science Technical Report*. No. MS-CIS-10-16. https://repository.upenn.edu/cgi/viewcontent.cgi?article=1970&context=cis_reports. 2010. 32 p.

24. Câmara, Daniel. "Formal Verification of Communication Protocols for Wireless Networks". *Publ. Belo Horizonte*, http://www.eurecom.fr/~camara/files/ThesisCamara_FormalVerification.pdf. Internet source. 2009. 136 p.

25. Gomes, Luís & Fernandes João M. "Behavioral Modeling for Embedded Systems and Technologies: Applications for Design and Implementation". 2010. 494 p. ISBN 13: 9781605667508 | ISBN 10: 1605667501 | EISBN13: 9781605667515 | DOI: https://doi.org/10.4018/978-1-60566-750-8.

26. Schamai, Wladimir. "Model-Based Verification of Dynamic System Behavior against Requirements Method, Language, and Tool". Linköping University SE-581 83. Internet source. Linköping: Sweden. 2013. 257 p.

27.Kudryavtsev, V. B., Grunskii, I. S. & Kozlovskii, V. A. "Analysis and Synthesis of Abstract Automata". *Journal of Mathematical Sciences*. 2010; Vol.169 Issue 4: 481–532. DOI: https://doi.org/10.1109/IDAACS.2019.8924314.

28. Martynyuk, O., Drozd, O., Ahmesh, Tamem, Bui Van Thuong, Sachenko, A., Mykhailova, H. & Dombrovskyi, M. "Hierachical Model of Behavior On-line Testing for Distributed Information Systems". *The 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*. Metz. France: 2019; Vol.2: 724–729.

29. Drozd, M., Drozd, O. & Antoniuk, V. V. "Power-oriented Checkability and Monitoring of the Current Consumption in FPGA Projects of the Critical Applications". *Scientific Journal Applied Aspects of Information Technology*. Odessa: Ukraine. 2019; Vol. 2 No 2: 105–114.

**Conflicts of Interest:** the authors declare no conflict of interest

Received 07.10.2019
Received after revision 26.11 2019
Accepted 03.12.2019

**DOI: https://doi.org/ 10.15276/aait.04.2019.4**
**УДК 004.738:004.94**

## ПОВЕДІНКОВА ВЕРИФІКАЦІЯ СИСТЕМ ІНТЕРНЕТУ РЕЧЕЙ НА ОСНОВІ МЕРЕЖ ПЕТРІ

**Олександр Миколайович Мартинюк**[1)]
ORCID: http://orcid.org/0000-0003-1461-2000, anmartynyuk@ukr.net, Scopus ID: 57103391900
**Олександр Валентинович Дрозд**[1)]
ORCID: https://orcid.org/0000-0002-9160-5982, drozd@ukr.net, Scopus ID: 55388226700
**Сергій Анатольевич Нестеренко**[1)]
ORCID: http://orcid.org/0000-0002-3053-0374, sa_nesterenko@ukr.net, Scopus ID: 55386373800
**Тамем Ахмеш**[1)]
ORCID: http://orcid.org/0000-0003-1461-2000, tamim.nor@yahoo.com
[1)] Одеський національний політехнічний університет, пр. Шевченка, 1. Одеса, 65044,Україна

## АНОТАЦІЯ

Швидкий розвиток, впровадження у всіх сферах людської діяльності та зростаюча відповідальність функцій Інтернет-систем речей посилюють і ускладнюють вимоги до надійності їх проектних рішень на етапах розробки та працездатності під час впровадження в життя. Загальновідомі методи перевірки проектів та реалізацій базуються на засобах системного, структурного, функціонального, конструкторсько-технологічного аналізу та синтезу Інтернет-систем речей. Однак їхні можливості не занижують доцільність розробки формалізованих моделей та методів перевірки. У цьому дослідженні представлені елементи технології та етапи методології перевірки поведінки проектів функціонального рівня для Інтернет-систем речей, представлених за допомогою мереж Петрі. Загальна перевірка представлена трьома етапами - аналіз правильності загальних структурних та функціональних властивостей, фактична перевірка міжрівневих та міжкомпонентних взаємодій, поведінкове тестування в режимі онлайн та офлайн у класі помилок функціонального типу. У запропонованому аналізі визначаються основні сутності та взаємозв'язки систем речей Інтернету речей та перевіряються архітектурний рівень, що визначає структуру, компоненти, функції, інтерфейси, взаємодії асинхронно-подій та представляють елементи мереж Петрі - їхні позиції, переходи, дуги , функції, розмітка. Тестування в Інтернеті та офлайн для динамічної перевірки поведінки в Інтернеті систем речей проводиться на основі відповідно фону або спеціального формування багатьох технологічних потоків в мережі Петрі, активованих під час її моделювання та охоплення об'єктів мережі Петрі . У цій роботі представлені загальні оцінки витрат ресурсів та часу на проектування Інтернет-систем речей без перевірки та з верифікацією, показано їх зменшення у разі помилок проектування, перероблення та застосування верифікації. Перевірка проілюстрована на прикладі мереж Петрі, що імітують автоматичну систему освітлення.

**Ключові слова:** системи Інтернет речей; поведінкова верифікація; мережа Петрі; покриття перевіряються властивостей; складність верифікації

**DOI: https://doi.org/ 10.15276/aait.04.2019.4**
**УДК 004.738:004.94**

## ПОВЕДЕНЧЕСКАЯ ВЕРИФИКАЦИЯ СИСТЕМ ИНТЕРНЕТА ВЕЩЕЙ НА ОСНОВЕ СЕТЕЙ ПЕТРИ

**Александр Николаевич Мартынюк**[1)]
ORCID: http://orcid.org/0000-0003-1461-2000, anmartynyuk@ukr.net, Scopus ID: 57103391900
**Александр Валентинович Дрозд**[1)]
ORCID: https://orcid.org/0000-0002-9160-5982, amberk4@gmail.com, Scopus ID: 55388226700
**Сергей Анатольевич Нестеренко**[1)]
ORCID: http://orcid.org/0000-0002-3053-0374, sa_nesterenko@ukr.net, Scopus ID: 55386373800
**Тамем Ахмеш**[1)]
ORCID: 0000-0003-1461-2000, lukianov@bsu.by
[1)] Одесский национальный политехнический университет, пр. Шевченко, 1. Одесса, 65044, Украина

## АННОТАЦИЯ

Бурное развитие, внедрение во все сферы человеческой деятельности и растущая ответственность функций систем Интернет вещей ужесточают и усложняют требования к надежности их проектных решений на этапах разработки и работоспособности при реализации внедрений. Известные методы проверки проектов и реализаций основаны на средствах системного, структурного, функционального, конструкторского и технологического анализа и синтеза систем Интернет вещей. Однако их возможности не недооценивают целесообразность разработки формализованных моделей и методов проверки. В этом исследовании представлены элементы технологии и этапы методологии поведенческой проверки проектов функционального уровня для систем Интернета вещей, представленных с использованием сетей Петри. Общая проверка представлена тремя этапами - анализ правильности общих структурных и функциональных свойств, фактическая проверка межуровневых и межкомпонентных взаимодействий, поведенческое онлайн и автономное тестирование в классе ошибок функционального типа. В предлагаемом анализе основные сущности и взаимосвязи систем Интернета вещей определяются и проверяются на архитектурном уровне, определяя структуру, компоненты, функции, интерфейсы, взаимодействия асинхронных событий и представляют элементы сетей Петри - их положения, переходы, дуги. , функции, разметка. Онлайновое и автономное тестирование для динамической проверки поведения систем интернета вещей осуществляется на основе, соответственно, фонового или специального формирования многих потоков процессов в сети Петри, активированных при ее моделировании и охватывающих объекты сети Петри. В этом документе представлены общие оценки затрат ресурсов и времени на проектирование систем Интернета вещей без проверки и с проверкой, показывающие их снижение в случае ошибок проектирования, перепроектирование и применение проверки. Проверка проиллюстрирована на примере сетей Петри, имитирующих автоматическую систему освещения.

**Ключевые слова:** системы Интернет вещей; поведенческая верификация; сеть Петри; покрытие проверяемых свойств; сложность верификации

## ABOUT THE AUTHORS

**Oleksandr N. Martynyuk,** PhD (Eng), Associate Professor of the Computer Intellectual Systems and Networks Department, Odessa National Polytechnic University. 1. Shevchenko Ave. Odesa, 65044, Ukraine
anmartynyuk@ukr.net. ORCID: http://orcid.org/0000-0003-2366-1920
***Research field*:** Behavioral Testing and Diagnosis of Computer Systems, Formal Verification of Project, Recognizing of Digital Systems

**Олександр Миколайович Мартинюк,** канд. техніч. наук, доцент каф. Комп'ютерних інтелектуальних систем і мереж. Одеський національний політехнічний університет, пр. Шевченка, 1. Одеса, 65044, Україна

**Oleksandr V. Drozd,** Dr. Sci. (Eng), Professor of the Computer Intellectual Systems and Networks Department, Odessa National Polytechnic University. 1, Shevchenko Ave. Odesa, 65044, Ukraine
drozd@ukr.net.ORCID: http://orcid.org/0000-0001-8305-2217
***Research field*:** Testing and Diagnosis of Computer Systems, Arithmetical Foundations of Computer Systems, Computer Systems and Components

**Олександр Валентинович Дрозд,**, д-р техніч. наук, проф. каф. Комп'ютерних інтелектуальних систем і мереж.Одеський національний політехнічний університет, пр. Шевченка, 1. Одеса, 65044, Україна

**Sergey A. Nesterenko,** Dr. Sci. (Eng), Professor of the Computer Intellectual Systems and Networks Department, Odessa National Polytechnic University. 1, Shevchenko Ave. Odesa, 65044, Ukraine
sa_nesterenko@ukr.net. ORCID: http://orcid.org/0000-0002-3053-0374
***Research field*:** Analyze, Modelling and Synthesis of Computer Systems and Networks

**Сергій Анатольевич Нестеренко,** д-р техніч. наук, проф. каф. Комп'ютерних інтелектуальних систем і мереж, проректор. Одеський національний політехнічний університет, пр. Шевченка, 1. Одеса, 65044, Україна

**Tamem Ahmesh,** Post-graduate Student of the Department of Computer Intellectual Systems and Networks,
tamim.nor@yahoo.com. ORCID: http://orcid.org/0000-0002-0482-2656.
Odessa National Polytechnic University, 1, Shevchenko Ave. Odessa, 65044, Ukraine.
***Research field*:** Testing and Diagnosis of Computer Systems, Computer Networks

**Тамем Ахмеш,** аспірант каф. Комп'ютерних інтелектуальних систем і мереж. Одеський національний політехнічний університет, пр. Шевченка, 1. Одеса, 65044, Україна