Applied problems of information systems operation

UDC 004.94 doi: https://doi.org/10.20998/2522-9052.2024.2.12

Oleg Barabash[1], Olha Svynchuk[1], Ivanna Salanda[2], Viktor Mashkov[3], Mykola Myroniuk[4]

[1] National Technical University of Ukraine “Igor Sikorsky KPI”, Kyiv, Ukraine
[2] Taras Shevchenko Regional Humanitarian and Pedagogical Academy of Kremenets, Kremenets, Ukraine
[3] University of J.E. Purkyne in Usti nad Labem, Czech Republic, Usti nad Labem, Czech Republic
[4] National Defense University of Ukraine, Kyiv, Ukraine

ENSURING THE FUNCTIONAL STABILITY OF THE INFORMATION SYSTEM OF THE POWER PLANT ON THE BASIS OF MONITORING THE PARAMETERS OF THE WORKING CONDITION OF COMPUTER DEVICES

Abstract. The functional stability of the information system of the power plant is ensured by a complex of processes and mechanisms that are capable of maintaining the normal operation of the system even in the event of errors, failures or negative impacts. **The aim of the research.** An important aspect of ensuring the functional stability of an information system is the monitoring of its healthy state, as it helps to identify, analyze and respond to any problems in a timely manner, ensuring the reliable and uninterrupted operation of the system. It was decided to choose a test diagnosis based on the principle of a wandering diagnostic core. **Research results.** An algorithm for detecting failures in the system has been developed based on the decryption of the totality of the results of the system's test checks. The developed software application allows you to monitor the state of various components of the information system and detect possible problems or failures in a timely manner in order to support the continuous operation of the system. This application allows you to increase the reliability of diagnostics, reduce the time of diagnostics, and carry out diagnostics with the specified completeness and depth. The depth and completeness of diagnosis is determined by the test task. **Verification.** To confirm the correctness of the developed software product, mathematical modeling of the process of diagnosing the information system, which was divided into several subsystems containing a certain number of modules, was carried out. For the division into subsystems, the number of modules in each subsystem is important - it should not exceed 30 modules. This limitation is due to the limited computing power of modern microprocessor technology during the solution of a class of NP-complete problems.

Keywords: information system; functional stability; external and internal destabilizing factors; monitoring; diagnostics; test control; power plant; probability.

Introduction

Reliable energy in the country is the main factor for the normal functioning of all spheres of its activity. Information systems of power plants function under the influence of external and internal destabilizing factors. Under negative influence, system modules may fail. However, the systems must function offline for a specified time. Such a condition of functioning can be fulfilled by ensuring the property of functional stability.

Functional stability is the possibility of functioning of the information system during the specified time under the influence of external and internal destabilizing factors [1]. External and internal destabilizing factors mean failures, failures of system modules, mechanical damage, errors of service personnel.

The main stages of ensuring functional stability are the detection of a module or several modules that have failed in the system, followed by their diagnosis and restoration of the functioning of the information system of the power plant.

To ensure the functional stability of the information system, the following procedures are performed:

– finding an abnormal situation and destabilizing influences that worsen the quality of the system's functioning;
– identification of identified destabilizing influences;
– making a decision to restore functioning system processes;
– recovery by redistribution of functions and tasks between intact modules.

Therefore, the process of controlling the working condition and monitoring the parameters of the functioning of multi-module information systems in the modern world is becoming a rather urgent task.

Literature review and problem statement

For the first time, the concept of functional stability was introduced by Professor O. Mashkov and further developed by O.Barabash for complex technical systems. The paper [2] describes the first stage of ensuring functional stability – timely and reliable control of the state of the information system: if the self-monitoring procedure signals the presence of a failure in the system, then the self-diagnosis procedure is already included to find the location of the failure. A number of scientific works by such scientists as O. Mashkov, O. Barabash, Yu. Kravchenko, D. Obidin, V. Sobchuk, A. Musienko are devoted to solving the problem of ensuring functional stability. and other. Solving the problems of functional stability of multi-module information systems responsible for critical infrastructure largely depends on the possibility of monitoring and forecasting their technical condition [3], determining the basic requirements and criteria for functionally stable complex systems [4].

The works [5] proposed methods for ensuring the fault tolerance of the information system and a method for diagnosing its malfunctions [6]. Based on the obtained results, new diagnostic models of the information system for detecting faulty processors are proposed [7].

The works [8, 9] describe the functional stability of rapidly developing systems. It has been found that these are rarely addressed and can lead to operational failure in real-world scenarios due to long recovery times after an adverse impact, functional fault tolerance testing prior to deployment will identify internal or injected vulnerabilities. The article [10] highlights different types of functionals characterizing the effectiveness of the information system, in conditions of incomplete a priori information about the distribution function of the determination of random variables, through which the main indicators of the reliability of the information system are expressed.

In [11], the methodology of building an effective system of self-diagnosis of information systems is described on the example of enterprises of the metallurgical and energy industries. Methods of organizing and performing self-diagnosis, detection mechanisms, as well as identification and localization of faulty modules are given. An analysis of the use of a hierarchical approach to the organization of means of ensuring functional stability was carried out [12]. Based on the results, algorithms were developed that form a two-level system for diagnosing hidden failures [13]. Features of the main provisions of the theory of artificial intelligence, namely neural networks to ensure functional stability of production processes of industrial enterprises are considered [14].

In the articles [15, 16], mechanisms of self-organization of heterogeneous information networks are disclosed, new criteria for determining functionally stable networks are proposed. Research was conducted on the functional stability of the structure of the information telecommunication network [17, 18] and the navigation support of civil aviation aircraft with the definition of new approaches [19]. Scientists consider various architectures of information systems from the point of view of functional stability, as well as its impact on network resources and network services [20, 21]. A dynamic VNF model of a computer epidemic has been improved, which allows predicting the level of functional stability of the information system at various stages of the epidemic [22].

Functionally stable ecologically complex systems and mathematical formalization of the properties of their functional stability are studied [23]. The methods of ensuring this property are aimed at more complete use of technical resources of technologically dangerous ecological systems [24]. The results of fundamental studies of the mechanisms of various types of corrosion, the impact of corrosion on the resistance of objects to the preservation and extension of operational suitability are presented [25].

For critical infrastructure objects, a method of constructing a law of safety management of critical infrastructure objects in the conditions of external uncontrolled influences is proposed [26, 27]. The security stability of information systems is described, where the security stability of information systems is defined as the dynamic ability of the system to respond to an attack and recover from it [28, 29]. In order to automate the system of control and diagnostics of microprocessor systems, it is proposed to implement the principle of self-diagnosis, which is based on the ideas of artificial intelligence [30].

Therefore, for all complex technical systems, it is very important to carry out continuous control of the parameters of functioning and good condition, since forecasting allows solving the problem of determining the optimal moments of control, in the intervals between which the property of functional stability of the system will be ensured. With the help of such control, it is possible to ensure an increase in the labor productivity of all modules while reducing the number of people employed in production and significantly reducing the share of manual labor.

There are various software solutions to ensure the reliability and productivity of information systems.

Nagios is a powerful tool that provides instant information about an organization's critical IT infrastructure [31]. Nagios monitors the health of servers, workstations, network devices, applications, services, various resources and allows you to detect and fix problems and mitigate future problems before they affect end users and customers, etc. Nagios uses a modular architecture that allows users to extend and customize the system based on their needs. Therefore, this software application allows you to create a powerful and flexible monitoring system that can be adapted to specific needs and expanded with the help of plugins and extensions.

SolarWinds Network Performance Monitor (NPM) is a powerful network monitoring software that allows you to quickly identify, diagnose, and resolve network performance issues and network failures [32]. The software product monitors the status and functional parameters of network devices, including routers, switches, firewalls, servers, and other assets, generates notifications in case of unusual conditions or network failures through various notification channels, etc. The architecture of the SolarWinds software solution is based on a distributed approach. The SolarWinds network monitoring and management system is a large software complex that provides extensive opportunities for monitoring and managing various aspects of the information infrastructure.

Having analyzed these software solutions, we can conclude that they provide the ability to monitor servers, applications, network devices and other resources, have a large community of users and expand with the help of plugins and extensions. However, none of these solutions provides the property of functional stability, which allows the system to work in offline mode for a given time.

The aim of this study is the development of an application for monitoring parameters of the serviceable state of computing devices of the information system of the power plant based on software-logical methods of test control to ensure functional stability.

Main part

The functional stability of the information system is ensured by a complex of processes and mechanisms that are capable of maintaining the normal operation of the system even in the event of errors, failures or negative impacts. An important aspect of ensuring the functional stability of an information system is the monitoring of its healthy state, as it helps to identify, analyze and respond to any problems in a timely manner, ensuring the reliable and uninterrupted operation of the system.

To achieve the functional stability of the information system, it is important to choose an appropriate system diagnosis algorithm that will take into account the features of the system and its modules. It will make it possible to determine whether the system modules are working properly, quickly detect malfunctions and take appropriate measures to ensure the reliability and performance of the IS.

Among the existing methods of diagnosis, the method of test diagnosis of components of the information system was chosen. The test method of diagnosis according to the construction principle is divided into two separate types: according to the principle of a centralized diagnostic core and a wandering diagnostic core. Test diagnostics with a centralized diagnostic core and test diagnostics with a wandering diagnostic core are two different approaches to conducting diagnostics in distributed information systems:

– test diagnostics with a centralized diagnostic core uses centralized control, that is, the diagnostic core is located in the central node of the system, which allows effective management and control of the diagnostic process;

– test diagnosis with a wandering diagnostic core distributes the diagnosis task between nodes, where the diagnostic functions are distributed among different system nodes, which helps to divide the load and increase productivity.

Both methods have their advantages and disadvantages, and the choice depends on the specific requirements and characteristics of the system. After comparing the two approaches of test diagnostics, it was decided that the organization of test diagnostics based on the principle of a wandering diagnostic core is the most acceptable.

The essence of the test diagnosis method with a wandering diagnostic core is as follows. Diagnostics consists of elementary checks by nodes of other nodes of the system, which are performed at random moments of time. Each node locally stores a matrix with the results of elementary tests in its memory. The exchange of diagnostic information about the results of checks is carried out between nodes based on the method of conditional transfer of the results of elementary checks. Each node, receiving diagnostic information, forms a sign of sufficiency with respect to the algorithm for deciphering the received diagnostic information. As a sign of sufficiency, checking the condition of reaching the limit of the number of elementary tests and checking whether all nodes have been tested is used. When the indicated sufficiency sign is satisfied, the node on which the sign is positively fulfilled performs the diagnostic information decryption algorithm and determines the technical condition of all nodes of the distributed information system. The process of determining the technical state of the node depends on conditional verification of the obtained result with the reliability of diagnosing the system. The diagnostic validity is user-defined or the default value is used [3].

The information system is represented by an undirected graph $G(V,E)$, $v_i \in V$, $e_{ij} \in E$, $i,j=\overline{1,n}$, which is described by the adjacency matrix. The set of vertices *V* corresponds to the set of working modules of dimension *n*, and to the set of edges *E* – the set of communication channels between modules. The information system performs the function of data exchange if there is at least one information transmission route between any pair of switching nodes. The graph connectivity requirement provides a reason to quantitatively assess the property of functional stability of the information system.

Let's consider the stages of the diagnosis algorithm.

Stage 1. Basic verification process.

Elementary checking of a node v_j from the side v_i is called submission of a test task t_{ij} to v_j and analysis of the node's reaction t'_{ij} to v_j the test task performed by the node v_i, where *i* is the number of the examiner, and *j* is the number of the node being checked.

An elementary check is as follows. The node v_i submits a test task to the node t_{ij}. The node v_j, having performed the test t_{ij}, sends a feedback response to the test t'_{ij} to the node v_i. The node v_i analyzes the reaction, compares it with the specified reference value and determines the result of elementary verification r_{ij}. The Preparata evaluation system was chosen for evaluation [7]:

$$r_{ij}=\begin{cases}0, \text{ if node } v_i \text{ is working, node } v_j \text{ is working;}\\ 1, \text{ if node } v_i \text{ is working, node } v_j \text{ is faulty;}\\ x=\{0\vee 1\}, \text{ if node } v_i \text{ is faulty.}\end{cases} \quad (1)$$

Stage 2. Transmission of diagnostic information.

The next main stage of the self-diagnosis algorithm is the transfer of diagnostic information based on the method of conditional transfer of the results of elementary checks. The checking node of the system, depending on the result of the check, can forward to the checked node or keep the diagnostic information. If the node v_i evaluates as working $\left(r_{ij}=0\right)$, it sends the results R_i accumulated by the node v_i to the node v_j. If node v_i evaluates the node v_j as faulty $\left(r_{ij}=1\right)$, then it does not send it, but keeps the results of checks v_i. The node v_j forwards the results of all previous checks R_j, to the node v_k if it evaluates it as working. With this method of information transmission, healthy nodes will receive much more diagnostic information than faulty ones.

Therefore, the syndrome decryption algorithm will be performed by one of the working nodes.

Stage 3. Storage of diagnostic information.

During the execution of the self-diagnosis algorithm, there is a need to store and transfer diagnostic information between nodes.

Diagnostic information is formed in the form of an *n* by *n* matrix R_{ij} (where *n* is the number of all nodes in the system, and *i* is the number of the node on which the matrix is stored), which consists of elementary checks in the system:

$$R_{ij} = \begin{pmatrix} r_{11} & \cdots & r_{nn} \\ \vdots & \ddots & \vdots \\ r_{n1} & \cdots & r_{nn} \end{pmatrix}. \quad (2)$$

Each node stores a check matrix R_i locally, which at the beginning of diagnosis consists of empty elements. With each check by a node v_i of other nodes of the system, the results of the checks r_{ij} are stored in its local matrix R_i. In addition to this option of replenishing the matrix with data, it can receive data as a result of merging with the matrix of another node v_j, provided that it is considered node v_i working $(r_{ji} = 0)$. Then $R_{ij} = R_i + R_j$.

Stage 4. Checking the sign of information sufficiency.

In order to determine the correct and faulty nodes of the system, it is necessary to check whether enough diagnostic information has been accumulated. To do this, when receiving the result of each elementary check, the node initiates a check of the sufficiency sign. If the sufficiency sign is fulfilled, the node generates a “stop” signal and proceeds to the execution of the last stage of the algorithm – deciphering the syndrome.

After the “stop” signal, all nodes clear their local result matrices and wait for the “start” signal to start a new self-diagnosis cycle.

The sign of sufficiency is characterized by two conditions:

– each node of the system was tested;

– he number of elementary checks is greater than or equal to the *limit* of checks limit, which is calculated by the formula:

$$limit = k \cdot n, \quad (3)$$

which *n* is the number of nodes in the system, *k* is adequacy ratio $k \in [1, n-1]$, which is specified by the user.

After each elementary check, the node checks these two conditions and, if they are fulfilled, goes to the last stage of the algorithm is decoding the syndrome. If at least one of the conditions was not met, the algorithm for accumulating diagnostic information (performing elementary checks) continues.

Stage 5. Deciphering the syndrome.

A node that has successfully checked the sufficiency sign starts deciphering the syndrome. Deciphering the syndrome consists in determining the posterior probabilities of the serviceability of each node, taking into account the obtained matrix of results (syndrome) *R* using the Bayes theorem. Each node has a known a priori probability of failure q_i.

Using the a priori probability of failure, it is possible to calculate the a priori probability of node serviceability $p_i = 1 - q_i$:

With the initial data in the form of a priori probabilities q_i, p_i and the obtained syndrome *R*, it is possible to calculate the a posteriori probabilities of serviceability of nodes p_i^*. Let's generate a set of possible hypotheses about the serviceability and malfunction of nodes in the following form of hypotheses, where 0 is the node is faulty, and 1 is the node is working:

$$\begin{aligned} &H_0 = \{0,0,0,...,0,0\}, \\ &H_1 = \{0,0,0,...,0,1\}, \\ &H_2 = \{0,0,0,...,1,0\}, \\ &.............................. \\ &H_n = \{1,1,1,...,1,1\}, \end{aligned} \quad (4)$$

which *n* is the number of possible hypotheses.

After obtaining all possible hypotheses of serviceability and malfunctions of nodes, it is necessary to calculate the probability of obtaining the syndrome, provided that the hypothesis is fulfilled $P(R/H_i),\ i = \overline{1,n}.$ We calculate the probabilities of each elementary test under the condition of the hypothesis being fulfilled, using the Preparata evaluation system (1). We receive:

$$P(R/H_i) = \prod_{e}^{|R|} P(r_e/H_i), \quad (5)$$

where *R* is the matrix of check results, *e* is an elementary check from the *R* matrix, $P(r_e/H_i)$ is the probability of obtaining the result of an elementary check r_e provided the hypothesis is fulfilled H_i.

The next step is to calculate the probabilities of hypotheses that have non-zero probabilities $P(R/H_i)$:

$$P(H_i) = \prod_{j}^{n} P_H(v_j), \quad (6)$$

where *n* is the number of nodes, $P_H(v_j)$ is the probability of the node v_j receiving a state of serviceability/failure according to the hypothesis H_i.

We calculate the full probability of getting the syndrome *P*(*R*):

$$P(R) = \sum_{i}^{k} P(H_i) \cdot P(R/H_i), \quad (7)$$

where *k* is the number of non-zero hypotheses H_i

According to Bayes' theorem, we determine the conditional probabilities of accepting the hypotheses H_i under the condition of obtaining the syndrome *R*:

$$P(H_i / R) = \frac{P(H_i) \cdot P(R / H_i)}{P(R)}. \qquad (8)$$

We calculate the posterior probabilities of node serviceability p_i^*. We determine the probabilities $P(H_i / R)$, where the node can be recognized as working after accepting the hypothesis H_i, which are independent, which makes it possible to apply the theorem on the addition of probabilities:

$$p_i^* = \sum P_{work}(H_i / R), \qquad (9)$$

which $P_{work}(H_i / R)$ is conditional probabilities of hypotheses $P(H_i / R)$, in which the node v_i is considered working.

Therefore, based on the found posterior probabilities, it is possible to draw a conclusion about the serviceability or malfunction of the nodes, comparing them with the diagnostic accuracy threshold *Acc*, which is set by the user. In case of detection of faulty nodes, it is necessary to localize the faulty nodes, distribute the load between the healthy ones and restore the operation of the distributed information system.

The software product is functionally divided into four blocks, communication between which is provided using HTTP requests and web sockets:

1) the monitoring panel (Web-GUI) is a web interface that provides the display of monitoring data to the user and the transfer of user requests to the next unit of the system;

2) master node is a separate node of the system, provides user interaction with the network of nodes, communication with the system database, communication of the web interface with the system and performs management of the network of system nodes;

3) database stores information about the status of system nodes over time, about events that occurred in the system, and about the results of each system diagnosis;

4) a network of nodes consists of elementary nodes that ensure the performance of tasks in the system, as well as perform a self-diagnosis algorithm for the health of themselves.

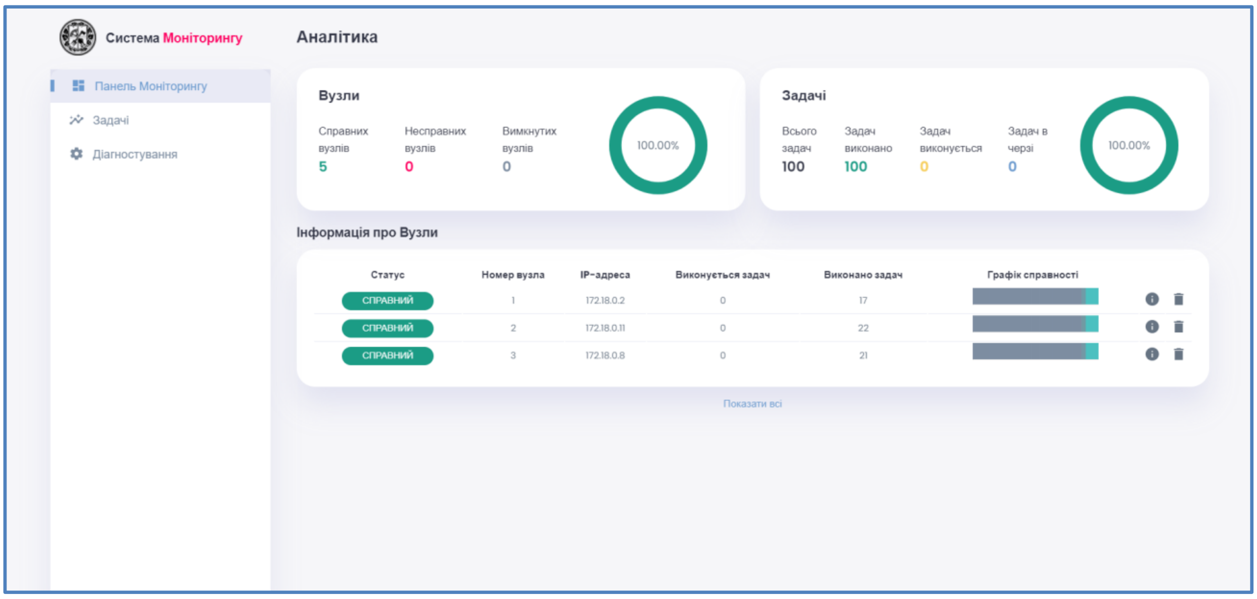
The developed software application makes it possible to monitor the state of various components of the information system and to detect possible problems or failures in a timely manner in order to support the continuous operation of the system by means of diagnostics based on software-logical methods of test control.

At the stage of operation, the information system operates in regular mode, during which it performs the functions assigned to it - the performance of a set of tasks.

The software application, part of which is embedded in the nodes of the distributed system, performs constant control of the system elements using the test method. In the event of a failure, damage or failure, the localization of the failed element is ensured, the system structure is changed by redistributing tasks to healthy nodes, information about the identified failures is issued to the system staff to initiate the repair process and restore the operational efficiency of the failed component of the distributed information system.

The monitoring panel is the main page of the program (Fig. 1).

This page contains general information about the system status of computing devices.



Справних вузлів	Несправних вузлів	Вимкнутих вузлів
5	0	0

Всього задач	Задач виконано	Задач виконується	Задач в черзі
100	100	0	0

Інформація про Вузли

Статус	Номер вузла	IP-адреса	Виконується задач	Виконано задач	Графік справності
СПРАВНИЙ	1	172.18.0.2	0	17	
СПРАВНИЙ	2	172.18.0.11	0	22	
СПРАВНИЙ	3	172.18.0.8	0	21	

Fig. 1. “Monitoring panel” menu

The page shows three main blocks of information:

– block about the serviceability of nodes in the system;

– a block on performing tasks in the system;

– a table with detailed information on each node of the system.

The block about the serviceability of nodes in the system provides information about the number of working, faulty and disabled nodes in the system.

A pie chart is built to the right of this information, which additionally presents the information in graphic form.

In the block about the execution of tasks in the system, there are four text fields that inform about the number of tasks in general, completed tasks, tasks that are being performed, and tasks in the queue. A pie chart is built to the right of this information, which additionally presents the information in graphic form.

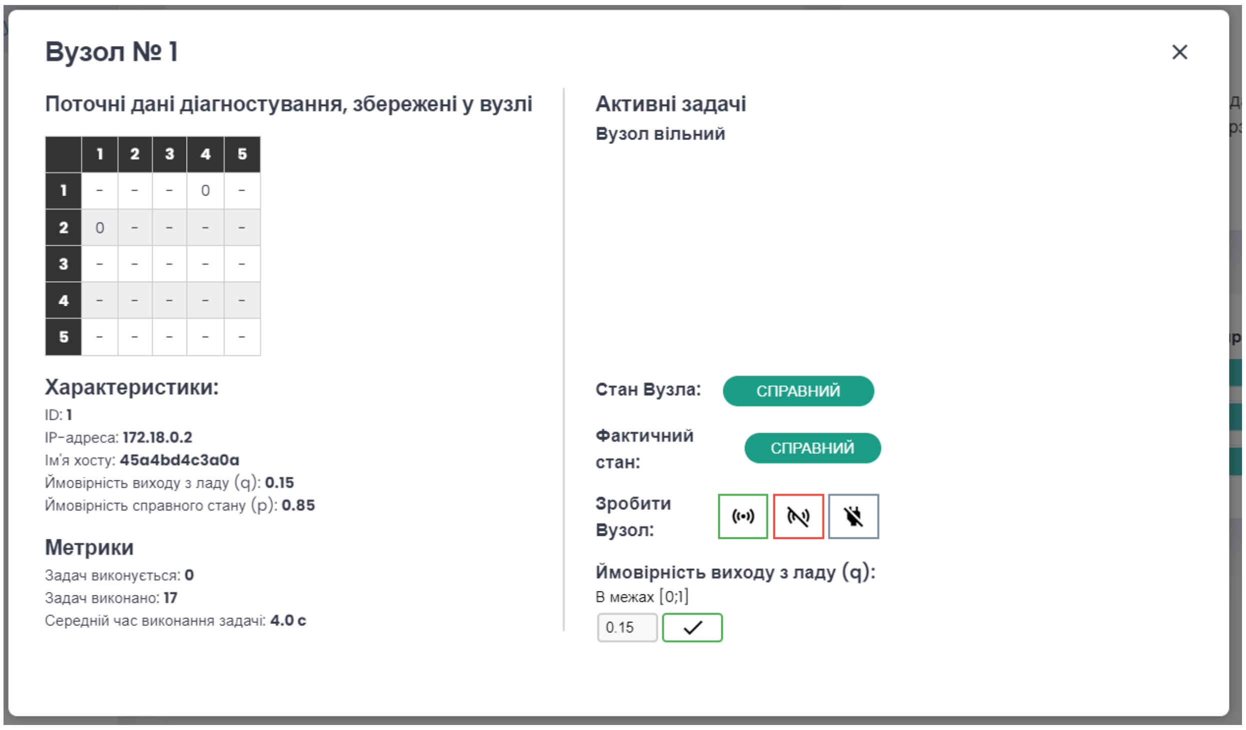
A table with detailed information of each node of the system informs about the serviceability or malfunction of a specific node, as well as additionally the following columns:

– node number (id);
– IP address of the node in the computer network;
– the number of tasks that are being performed right now;
– the total number of completed tasks by this node;
– graph of the state of the node for the last 30 minutes;
– "Details" button about this node;
– node deletion button.

When you click on the "Details" button, a window with detailed information about the node opens on top of the main page (Fig. 2).

The window is divided into two parts. Information about the node is displayed in the left part:

– the local matrix of diagnostic results for the current time, stored in the memory on the node;
– node number (id);
– IP address of the node in the computer network;
– the name of the host in the network (hostname);
– node failure probability (*q*);
– node serviceability probability (*p*);
– the number of tasks that are being performed right now;
– the total number of completed tasks by this node;
– the average time of execution of a task by a node (in seconds).


Вузол № 1

Поточні дані діагностування, збережені у вузлі

	1	2	3	4	5
1	-	-	-	0	-
2	0	-	-	-	-
3	-	-	-	-	-
4	-	-	-	-	-
5	-	-	-	-	-

Характеристики:
ID: **1**
IP-адреса: **172.18.0.2**
Ім'я хосту: **45a4bd4c3a0a**
Ймовірність виходу з ладу (q): **0.15**
Ймовірність справного стану (p): **0.85**

Метрики
Задач виконується: **0**
Задач виконано: **17**
Середній час виконання задачі: **4.0 c**

Активні задачі
Вузол вільний

Стан Вузла: СПРАВНИЙ
Фактичний стан: СПРАВНИЙ
Зробити Вузол:
Ймовірність виходу з ладу (q):
В межах [0;1]
0.15 ✓

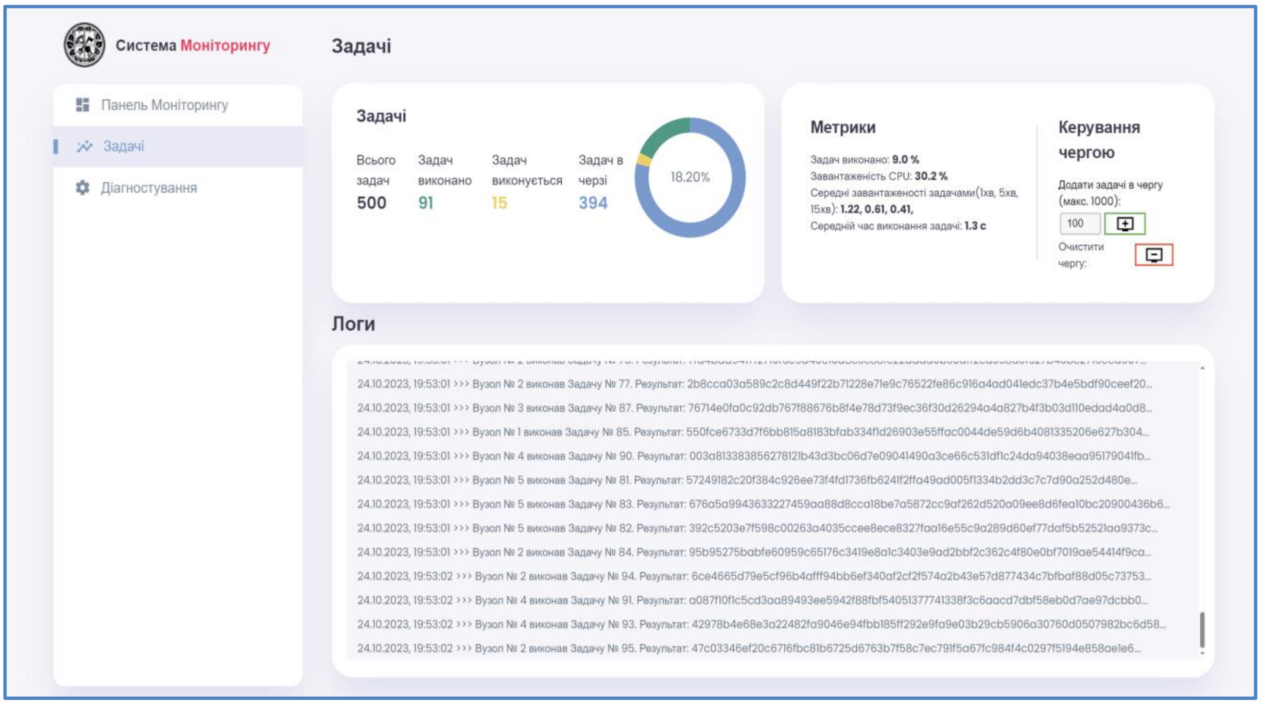
Fig. 2. Node information window

On the right side, information about the execution of tasks by the node is displayed: if the node is not performing tasks, then the inscription "Node is free" is displayed, if the node is performing tasks, then it shows the unique number of the task and the text that the node encrypts. After the "Active Tasks" section, there are two fields that indicate the health status of the node: "Node Status" shows the state diagnosed by the algorithm, and "Actual Status" shows the state of the node, which only the node itself knows.

The Make Node box has three buttons that change the actual state of the node to working, failed, and disabled, respectively. You can also change the probability of node failure. Clicking on the "Remove" button of the corresponding node will remove this node from the system network.

The "Tasks" tab contains all the functionality related to the logic of performing tasks in the system (Fig. 3).

On this tab, the block about the execution of tasks in the system is duplicated, and a new block is also placed, which displays more detailed information about the execution, namely: about the progress of the execution of tasks, the load on the server processor, the average load for 1, 5 and 15 minutes (Load Average) and the average task execution time of all nodes together.



Всього задач	Задач виконано	Задач виконується	Задач в черзі
500	91	15	394

Задач виконано: **9.0 %**
Завантаженість CPU: **30.2 %**
Середні завантаженості задачами(1хв, 5хв, 15хв): **1.22, 0.61, 0.41,**
Середній час виконання задачі: **1.3 с**

Додати задачі в чергу (макс. 1000): 100
Очистити чергу:

```
24.10.2023, 19:53:01 >>> Вузол № 2 виконав Задачу № [illegible]. Результат: [illegible]
24.10.2023, 19:53:01 >>> Вузол № 2 виконав Задачу № 77. Результат: 2b8cca03a589c2c8d449f22b71228e71e9c76522fe86c916a4ad041edc37b4e5bdf90ceef20...
24.10.2023, 19:53:01 >>> Вузол № 3 виконав Задачу № 87. Результат: 76714e0fa0c92db767f88676b8f4e78d73f9ec36f30d26294a4a827b4f3b03d110edad4a0d8...
24.10.2023, 19:53:01 >>> Вузол № 1 виконав Задачу № 85. Результат: 550fce6733d7f6bb815a8183bfab334f1d26903e55ffac0044de59d6b4081335206e627b304...
24.10.2023, 19:53:01 >>> Вузол № 4 виконав Задачу № 90. Результат: 003a81338385627812 1b43d3bc06d7e09041490a3ce66c531df1c24da94038eaa9517904 1fb...
24.10.2023, 19:53:01 >>> Вузол № 5 виконав Задачу № 81. Результат: 57249182c20f384c926ee73f4fd1736fb6241f2ffa49ad005f1334b2dd3c7c7d90a252d480e...
24.10.2023, 19:53:01 >>> Вузол № 5 виконав Задачу № 83. Результат: 676a5a9943633227459aa88d8cca18be7a5872cc9af262d520a09ee8d6fea10bc20900436b6...
24.10.2023, 19:53:01 >>> Вузол № 5 виконав Задачу № 82. Результат: 392c5203e7f598c00263a4035ccee8ece8327faa16e55c9a289d60ef77daf5b52521aa9373c...
24.10.2023, 19:53:01 >>> Вузол № 2 виконав Задачу № 84. Результат: 95b95275babfe60959c65176c3419e8a1c3403e9ad2bbf2c362c4f80e0bf7019ae54414f9ca...
24.10.2023, 19:53:02 >>> Вузол № 2 виконав Задачу № 94. Результат: 6ce4665d79e5cf96b4afff94bb6ef340af2cf2f574a2b43e57d877434c7bfbaf88d05c73753...
24.10.2023, 19:53:02 >>> Вузол № 4 виконав Задачу № 91. Результат: a087f10f1c5cd3aa89493ee5942f88fbf54051377741338f3c6aacd7dbf58eb0d7ae97dcbb0...
24.10.2023, 19:53:02 >>> Вузол № 4 виконав Задачу № 93. Результат: 42978b4e68e3a22482fa9046e94fbb185ff292e9fa9e03b29cb5906a30760d0507982bc6d58...
24.10.2023, 19:53:02 >>> Вузол № 2 виконав Задачу № 95. Результат: 47c03346ef20c6716fbc81b6725d6763b7f58c7ec791f5a67fc984f4c0297f5194e858ae1e6...
```

Fig. 3. "Tasks" menu

Average load shows the demand for running threads as the average number of running and waiting threads:

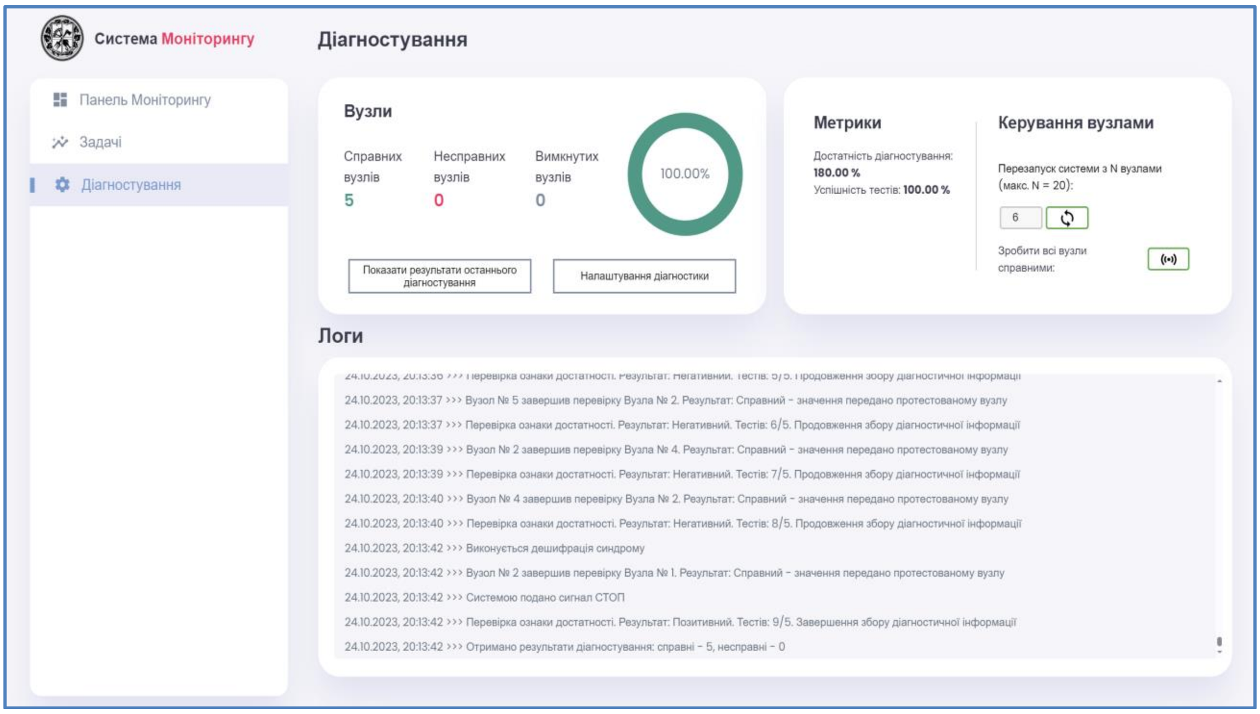
– if the values are equal to 0.0, then the system is idle;

– if the average value for 1 minute is higher than for 5 or 15, the load increases;

– if the average value for 1 minute is lower than for 5 or 15, the load is reduced;

– if the load value is higher than the number of processors, then you may have performance problems (depending on the situation).

The "Diagnostics" menu displays information about the algorithm for diagnosing system nodes (Fig. 4).



Справних вузлів	Несправних вузлів	Вимкнутих вузлів
5	0	0

Достатність діагностування: **180.00 %**
Успішність тестів: **100.00 %**

Перезапуск системи з N вузлами (макс. N = 20): 6
Зробити всі вузли справними:

```
24.10.2023, 20:13:36 >>> Перевірка ознаки достатності. Результат: Негативний. Тестів: 5/5. Продовження збору діагностичної інформації
24.10.2023, 20:13:37 >>> Вузол № 5 завершив перевірку Вузла № 2. Результат: Справний – значення передано протестованому вузлу
24.10.2023, 20:13:37 >>> Перевірка ознаки достатності. Результат: Негативний. Тестів: 6/5. Продовження збору діагностичної інформації
24.10.2023, 20:13:39 >>> Вузол № 2 завершив перевірку Вузла № 4. Результат: Справний – значення передано протестованому вузлу
24.10.2023, 20:13:39 >>> Перевірка ознаки достатності. Результат: Негативний. Тестів: 7/5. Продовження збору діагностичної інформації
24.10.2023, 20:13:40 >>> Вузол № 4 завершив перевірку Вузла № 2. Результат: Справний – значення передано протестованому вузлу
24.10.2023, 20:13:40 >>> Перевірка ознаки достатності. Результат: Негативний. Тестів: 8/5. Продовження збору діагностичної інформації
24.10.2023, 20:13:42 >>> Виконується дешифрація синдрому
24.10.2023, 20:13:42 >>> Вузол № 2 завершив перевірку Вузла № 1. Результат: Справний – значення передано протестованому вузлу
24.10.2023, 20:13:42 >>> Системою подано сигнал СТОП
24.10.2023, 20:13:42 >>> Перевірка ознаки достатності. Результат: Позитивний. Тестів: 9/5. Завершення збору діагностичної інформації
24.10.2023, 20:13:42 >>> Отримано результати діагностування: справні – 5, несправні – 0
```

Fig. 4. "Diagnostics" menu

On this tab, the block about working nodes in the system is duplicated, and a new block is also placed that displays more detailed information about diagnostics.

Two buttons appeared in the block about healthy nodes: "Show the results of the last diagnosis" and "Diagnosis settings".

When you click on the “Show results of the last diagnosis” button, a window opens with information about the last result of the diagnosis algorithm – the time when such a result was obtained, a matrix of the results of checks and the probability of serviceability of each node.

When you click on the “Diagnostics settings” button, another pop-up window opens, which displays the functionality of changing system diagnostics parameters:

– diagnostic accuracy;
– adequacy ratio;
– probability of failure of all nodes (one button for all).

In the “Node Management” part of the block, you can restart the system with a certain number of nodes in the network.

Also, for convenience, there is a button that changes the actual state of all nodes to working with one click.

Below, under the blocks, there are logs – a log of events for the last 5 minutes related to the diagnosis algorithm.

The log stores the time of the event and what kind of event occurred:

– the node has performed a basic check;
– verification of the sign of sufficiency;
– result of diagnosis;
– execution of syndrome decipherment;
– “start” signal is given;
– a “stop” signal is given.

To confirm the correctness of the developed software product, mathematical modeling of the process of diagnosing the information system was carried out (Fig. 5).

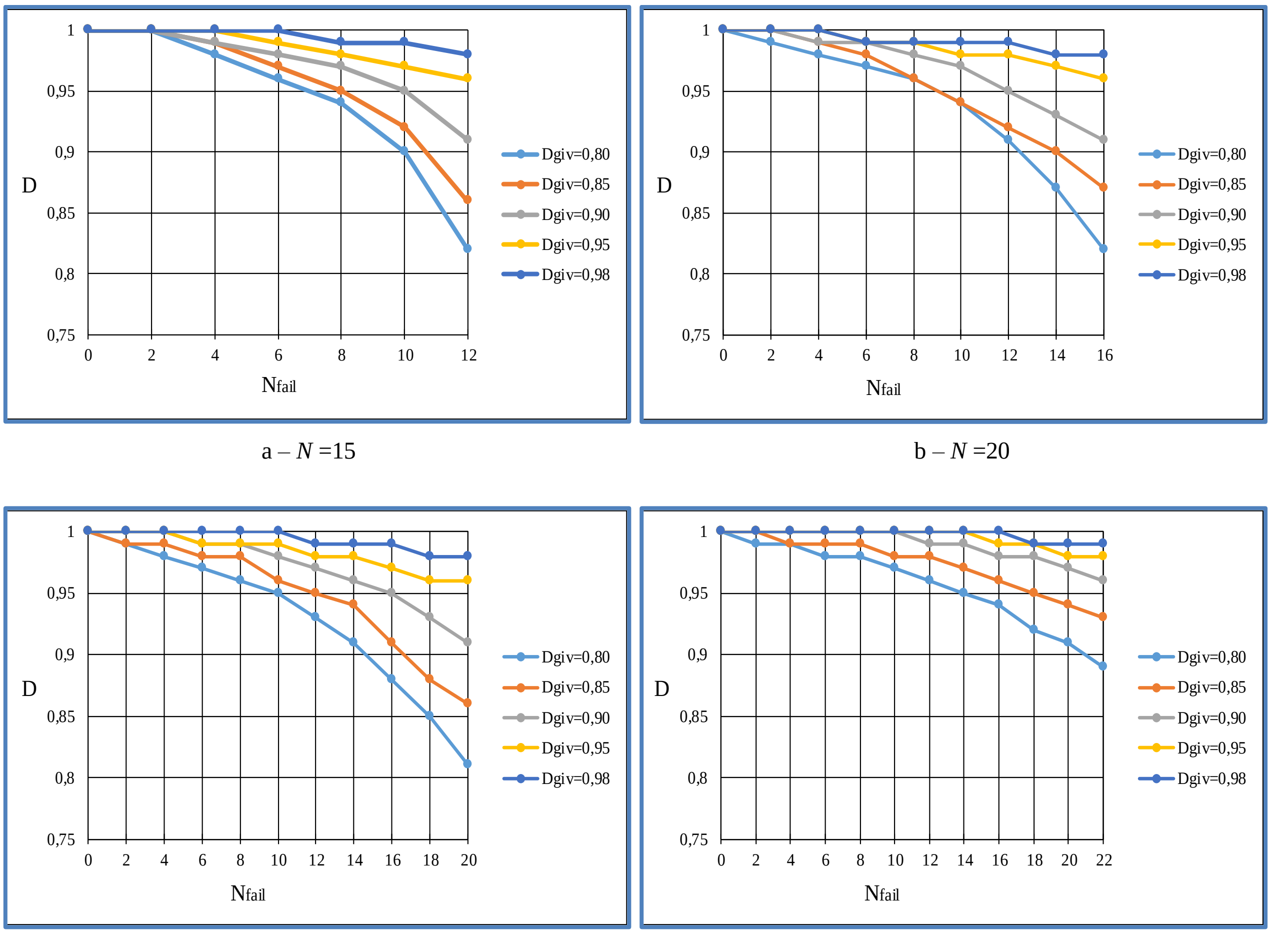


Fig. 5. Dependence of the reliability of diagnosing D on the number of failed modules N_{fail} =0,95

At the same time, the information system was divided into several subsystems, which contain a certain number of modules.

For the division into subsystems, the number of modules in each subsystem is important – it should not exceed 30 modules.

This limitation is due to the limitation of the computing power of modern microprocessor technology when solving a class of NP-complete problems.

Based on the results of the simulation, the change in the reliability of diagnosis based on the number of modules that failed at different values of the given level of reliability was determined D_{giv} and p.

Analysis of these graphs shows that if there are no failed modules N_{fail} =0, in all cases there will be 100 percent reliability D=1:

– at N_{fail} <5 the reliability of diagnosis approaches 1: D≥0,98. Subject to a significant number of rejections

N_{fail} = 6 … 12 for a system with N=15 modules, the reliability of diagnosis in any conditions is higher than D_{giv};

– at N_{fail} <7 the reliability of diagnosis approaches 1: D≥0,98. Subject to a significant number of rejections N_{fail} = 8 … 16 for a system with N=20 modules, the reliability of diagnosis in any conditions is higher than D_{giv};

– at $N_{fail} < 9$ the reliability of diagnosis approaches 1: $D \geq 0,98$. Subject to a significant number of rejections N_{fail} = 10 … 20 for a system with N = 25 modules, the reliability of diagnosis in any conditions is higher than D_{giv};

– at N_{fail} <11 the reliability of diagnosis approaches 1: $D \geq 0,95$. Subject to a significant number of rejections N_{fail} = 14 … 22 for a system with N=30 modules, the reliability of diagnosis in any conditions is higher than D_{giv}.

The analysis of the graphs of the dependence of the reliability of diagnosis D on the number of failures in the N_{fail} subsystem under the condition of varying the dimensions of the subsystem N and the a priori probability of the serviceable state of the modules p and the given reliability D_{giv} allows us to draw conclusions:

– the reliability of diagnostics practically does not change due to the expansion of the system (change in the number of system modules N);

– reliability values significantly depend on the number of failures N_{fail} – when the number of N_{fail} increases, the reliability of diagnosis decreases D;

– the dependence of the reliability on the a priori probability of the working state of modules p shows a slight increase in reliability with increasing value p.

Therefore, these results confirm the main property of test diagnosis – the possibility of diagnosis with a reliability not lower than the specified one.

Conclusions

A software application has been created for monitoring parameters of the serviceable state of computing devices of the information system of the power plant based on the self-diagnosis algorithm using a test method with a wandering diagnostic core to ensure functional stability. An algorithm for detecting failures in the system has been developed based on the decryption of the totality of the results of the system's test checks. This application allows you to increase the reliability of diagnostics, reduce the time of diagnostics, and carry out diagnostics with the specified completeness and depth. The depth and completeness of diagnosis is determined by the test task.

To confirm the correctness of the developed software product, mathematical modeling of the process of diagnosing the information system, which was divided into several subsystems containing a certain number of modules, was carried out. For the division into subsystems, the number of modules in each subsystem is important - it should not exceed 30 modules. This limitation is due to the limited computing power of modern microprocessor technology during the solution of a class of NP-complete problems.

Prospects for further research are seen in the improvement of the system due to the addition of new functionality, namely the introduction of artificial intelligence to ensure the functional stability of the information system of the power plant.

References

1. Sobchuk, V.V., Barabash, O.V. and Musijenko, A.P. (2022), *Basics of ensuring the functional stability of information systems of enterprises under the influence of destabilizing factors*, monograph, Milenium, Kyiv, 272 p., available at: https://www.researchgate.net/publication/363474851_Basis_for_functional_stability_of_information_systems_businesses_under_the_influence_of_destabilizing_factors
2. Mashkov, V.A. and Barabash, O.V. (1995), “Self-Checking of modular systems under random performance of elementary checks”, *Engineering Simulation*, Vol. 12, pp. 433–445.
3. Barabash, O., Sobchuk, V., Musienko, A., Laptiev, O., Bohomia, V. and Kopytko, S. (2023), “System Analysis and Method of Ensuring Functional Sustainability of the Information System of a Critical Infrastructure Object”, *System Analysis and Artificial Intelligence. Studies in Computational Intelligence*, vol. 1107, pp. 117–192, doi: https://doi.org/10.1007/978-3-031-37450-0_11
4. Barabash, O.V. (2004), *Construction of functionally stable distributed information systems*, NAOU, Kyiv, 226 p., available at: https://bit.ly/3wM5tDL
5. Peng, S.-L., Lin, C.-K., Tan, J.J.M. and Hsu, L.-H. (2012), “The g-Good-Neighbor Conditional Diagnosability of Hypercube under PMC Model”, *Applied Mathematics and Computation*, vol. 218, no. 21, pp. 10406–10412, doi: https://doi.org/10.1016/j.amc.2012.03.092
6. Yuan, J., Liu, A., Ma, X., Liu, X., Qin, X. and Zhang, J. (2015), “The g-Good-Neighbor Conditional Diagnosability of k-Ary n-Cubes under the PMC Model and MM Model”, *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 1165–1177, doi: http://doi.org/10.1109/TPDS.2014.2318305
7. Ren, Y. and Wang, S. (2016), “Some Properties of the g-Good-Neighbor (g-Extra) Diagnosability of a Multiprocessor Syste”, *American Journal of Computational Mathematics*, vol. 6, no. 3, pp. 259–266, doi: http://doi.org/10.4236/ajcm.2016.63027
8. Rohret, D., Kraft, M. and Vella, M. (2013),” Functional Resilience, Functional Resonance and Threat Anticipation for Rapidly Developed Systems”, *Journal of Information Warfare*, vol. 12, no. 2, pp. 50–62, doi: https://www.jstor.org/stable/26486855
9. Bellini, E., Coconea, L. and Nesi, P. (2020), “A Functional Resonance Analysis Method Driven Resilience Quantification for Socio-Technical Systems”, *IEEE Systems Journal*, vol. 14, no. 1, pp. 1234–1244, doi: http://doi.org/10.1109/JSYST.2019.2905713
10. Raskin, L., Karpenko, V., Ivanchykhin, Y. and Sokolov, D. (2023), “Diagnosis of systems under conditions of small initial data sampling”, *Advanced Information Systems*, vol. 7, no.3, pp. 39–43, doi: https://doi.org/10.20998/2522-9052.2023.3.05
11. Sobchuk, A.V. and Olimpijeva, Ju.I. (2020), “Application of neural networks to ensure functional stability of production processes”, *Telecommunication and Information Technologies*, vol. 2(67), pp. 13–28, doi: http://doi.org/10.31673/2412-4338.2020.021328

12. Sobchuk, V., Barabash, O., Musienko, A. and Svynchuk, O. (2021), "Adaptive accumulation and diagnostic information systems of enterprises in energy and industry sectors", *E3S Web of Conferences*, vol. 250, pp. 82–87, doi: https://doi.org/10.1051/e3sconf/202125008002
13. Sobchuk, V.V., Musijenko, A.P. and Iljin, O.Ju. (2018), "Analysis of the use of a hierarchical structure to ensure the functional stability of an automated enterprise management system", *Telecommunication and Information Technologies*, vol. 4 (61), pp. 53–61, doi: http://doi.org/10.31673/2412-4338.2018.045361
14. Sobchuk, V.V., Kovalj, M.O., Musijenko, A.P. and Macjko O.J. (2019), "The method of diagnosing hidden failures in the information system based on the application of a two-level system for ensuring functional stability", *Telecommunication and Information Technologies*, vol. 1 (62), pp. 22–31, doi: http://doi.org/10.31673/2412-4338.2019.012230
15. Maksymuk, O.V., Sobchuk, V.V., Salanda, I.P. and Sachuk, Yu.V. (2020), "A system of indicators and criteria for evaluation of the level of functional stability of information heterogenic networks", *Mathematical Modeling And Computing*, vol. 7, no. 2, pp. 285–292, doi: http://doi.org/10.23939/mmc2020.02.285
16. Kovalenko, A. and Kuchuk, H. (2022), "Methods to Manage Data in Self-healing Systems", *Studies in Systems, Decision and Control*, vol. 425, pp. 113–171, doi: https://doi.org/10.1007/978-3-030-96546-4_3
17. Obidin, D.M. (2014), "Assessment of functional stability of information and telecommunication networks based on automated control systems, *Nauka i tekhnika Povitrjanykh Syl Zbrojnykh Syl Ukrajiny*, vol. 1(40), pp. 167–169, available at: http://nbuv.gov.ua/UJRN/Nitps_2014_1_40
18. . Kuchuk, N., Mozhaiev, O., Semenov, S., Haichenko, A., Kuchuk, H., Tiulieniev, S., Mozhaiev, M., Davydov, V., Brusakova, O. and Gnusov, Y. (2023), "Devising a method for balancing the load on a territorially distributed foggy environment", *Eastern-European Journal of Enterprise Technologies*, vol. 1(4 (121), pp. 48–55, doi: https://doi.org/10.15587/1729-4061.2023.274177
19. Kalashnyk, Gh.A., Obidin, D.M. and Kalashnyk, M.A. (2016), "Ensuring stable functioning of aircraft navigation aids under the influence of external destabilizing factors", *Systemy obrobky informaciji*, vol. 3 (140), pp. 52–56, available at: http://nbuv.gov.ua/UJRN/soi_2016_3_15
20. Dovgiy, S., Kopiika, O. and Kozlov, O. (2021), "Architectures for the Information Systems, Network Resources and Network Services", *CEUR Workshop Proceedings: CPITS-II-1: Cybersecurity Providing in Information and Telecommunication Systems II*, vol. 3187, pp. 293–301, available at: https://ceur-ws.org/Vol-3187
21. Petrovska, I. and Kuchuk, H. (2023), "Adaptive resource allocation methodfor data processing and security in cloud environment", *Advanced Information Systems*, vol. 7, no. 3, pp. 67–73, doi: https://doi.org/10.20998/2522-9052.2022.3.10
22. Bychkov, A., Dimitrov, G., Shevchenko, V. and Shevchenko A. (2020), "Perfection of Computer Epidemic Model by Estimation of Functional Stability of the Information System", *Journal of Automation and Information Sciences*, vol. 52, no.1, pp. 29–40, doi: http://doi.org/10.1615/JAutomatInfScien.v52.i1.40
23. Mashkov, O., Chumakevych, V., Sokulsky, O. and Chyrun L. (2019), "Features of determining controlling effects in functionally-stablesystems with the recovery of a control", *Mathematical Modeling And Computing*, vol. 6, no. 1, pp. 85–91, doi: http://doi.org/10.23939/mmc2019.01.085
24. Mashkov, O.A., Alj-Tamymy, R.K.N., Lamy, D.D.Kh. and Kosenko V.R. (2016), "Functional stability of complex ecologically hazardous dynamic systems", *Ekologhichni nauky*, no. 14–15, pp. 65–74, doi: http://www.ecoj.dea.kiev.ua/archives/2016/14-15/10.pdf
25. Mashkov, O.A., Chumakevich, V.A., Mamchur, Yu.V. and Kosenko V.R. (2020), "The method of inverse problems of dynamics for the synthesis of a system of stabilization of the movement of a dynamic object on operatively programmable trajectories", *Mathematical Modeling And Computing*, Vol. 7, No. 1, pp. 29–38, doi: http://doi.org/10.23939/mmc2020.01.029
26. Mozhaev, O., Kuchuk, H., Kuchuk, N., Mykhailo, M. and Lohvynenko, M. (2017), "Multiservice network security metric", *2nd International Conference on Advanced Information and Communication Technologies*, AICT 2017 – Proceedings, pp. 133–136, doi: https://doi.org/10.1109/AIACT.2017.8020083
27. Laptiev, O., Barabash, O., Tsyganivska, I., Obidin, D. and Sobchuk, A. (2023), "The Method of Construction of the Law of Safety Management of Critical Infrastructure Objects Under the Conditions of External Uncontrolled Influences", *CEUR Workshop Proceedings*, vol. 3624, pp. 291–300, available at: https://ceur-ws.org/Vol-3624/Paper_24.pdf
28. Semenov, S., Sira, O., Kuchuk, N. (2018), "Development of graphicanalytical models for the software security testing algorithm", *Eastern-European Journal of Enterprise Technologies*, vol 2, no 4 (92), pp. 39–46, doi: https://doi.org/10.15587/1729-4061.2018.127210
29. Goel, L., Russell, D., Williamson, S. and Zhang, J.Z. (2023), "Information systems security resilience as a dynamic capability", *Journal of Enterprise Information Management*, vol. 36, no. 4, pp. 906–924, doi: https://doi.org/10.1108/JEIM-07-2022-0228
30. Tjuljupa, S.V., Samokhvalov, Ju.Ja., Khusainov, P.V. and Shtatenko S.S. (2023), "Self-diagnosis as a way to increase the cyber resistance of the terminal components of the technological system", *Cybersecurity: Education, Science, Technique*, Electronic Professional Scientific Edition, vol. 2, no. 22, pp. 134–147, doi: https://doi.org/10.28925/2663-4023.2023.22.134147
31. (2024), "The Open Source Standard In Monitoring", *Nagious Open Source*, available at: https://www.nagios.org/
32. (2024), "Network-performance-monitor", *Network Monitoring System*, SolarWinds" available at: https://www.solarwinds.com/network-performance-monitor

Надійшла (received) 22.02.2024
Прийнята до друку (accepted for publication) 15.05.2024

Відомості про авторів/ About the Authors

Барабаш Олег Володимирович, доктор технічних наук, професор, професор кафедри інженерії програмного забезпечення в енергетиці, Національний технічний університет України «Київський політехнічний університет імені Ігоря Сікорського», Київ, Україна;
Oleg Barabash, Doctor of Technical Sciences, Professor, Professor of the Department of Software Engineering in Energy System, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic University", Kyiv, Ukraine;

e-mail: bar64@ukr.net; ORCID ID: https://orcid.org/0000-0003-1715-0761;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=36724076700.

Свинчук Ольга Василівна, кандидат фізико-математичних наук, доцент, доцент кафедри інженерії програмного забезпечення в енергетиці, Національний технічний університет України «Київський політехнічний університет імені Ігоря Сікорського, Київ, Україна;
Olha Svynchuk, Candidate of Physical and Mathematical Sciences, Associate Professor, Associate Professor of the Department of Software Engineering in Energy System. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic University", Kyiv, Ukraine;
e-mail: 7011990@ukr.net; ORCID ID: https://orcid.org/0000-0001-9032-6335;
ScopusID: https://www.scopus.com/authid/detail.uri?authorId=57209510877.

Саланда Іванна Петрівна, кандидат технічних наук, доцент, доцент кафедри математики та природничих дисциплін, Кременецька обласна гуманітарно-педагогічна академія імені Тараса Шевченка, Кременець, Україна;
Ivanna Salanda, Candidate of Technical Sciences, Associate Professor, Associate Professor of the Department Mathematics and Natural Sciences, Taras Shevchenko Regional Humanitarian and Pedagogical Academy of Kremenets, Kremenets, Ukraine;
e-mail: salanda.ivanna@gmail.com; ORCID ID: https://orcid.org/0000-0002-5697-8564;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57216149027.

Машков Віктор Альбертович, доктор технічних наук, доцент, доцент кафедри інформаційних технологій, Університет Яна Евангелісти Пуркіне в Усті-над-Лабем, Усті-над-Лабем, Чеська Республіка;
Viktor Mashkov, Doctor of Technical Sciences, Associate Professor, Professor of the Department of Information Technology, University of J.E. Purkyne in Usti nad Labem, Usti nad Labem, Czech Republic;
e-mail: victor.mashkov@ujep.cz; ORCID ID: https://orcid.org/0000-0001-9817-3388;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=56962713100.

Миронюк Микола Юрійович, кандидат технічних наук, начальник науково-дослідного відділу, Національний університет оборони України, Київ, Україна;
Mykola Myroniuk, Candidate of Technical Sciences, Head of the Research Department, National Defense University of Ukraine, Kyiv, Ukraine;
e-mail: ycpex83@gmail.com; ORCID ID: https://orcid.org/0000-0002-7164-2700;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57226640271.

Забезпечення функціональної стійкості інформаційної системи електростанції на основі моніторингу параметрів справного стану обчислювальних пристроїв

О. В. Барабаш, О. В. Свинчук, І. П. Саланда, В. А. Машков, М. Ю. Миронюк

Анотація. Функціональна стійкість інформаційної системи електростанції забезпечується комплексом процесів та механізмів, які здатні підтримувати нормальну роботу системи навіть в умовах виникнення помилок, відмов або негативних впливів. **Мета дослідження.** Важливим аспектом забезпечення функціональної стійкості інформаційної системи є моніторинг її справного стану, оскільки він допомагає ідентифікувати, аналізувати та вчасно реагувати на будь-які проблеми, забезпечуючи надійну та безперебійну роботу системи. Було прийнято рішення обрати тестове діагностування за принципом блукаючого діагностичного ядра. **Результати дослідження.** Розроблено алгоритм виявлення відмов в системі на основі дешифрації сукупності результатів тестових перевірок системи. Розроблений програмний застосунок дозволяє відстежувати стан різних компонентів інформаційної системи та своєчасно виявляти можливі проблеми або відмови з метою підтримки безперервної роботи системи. Даний застосунок дозволяє підвищити достовірність діагностування, знизити час діагностування та проводити діагностування із заданою повнотою та глибиною. Глибина і повнота діагностування визначаються тестовим завдання. **Верифікація.** Для підтвердження правильності розробленого програмного продукту проведено математичне моделювання процесу діагностування інформаційної системи, яка розбивалась на кілька підсистем, які містять в собі певне число модулів. Для розбиття на підсистеми важливе значення має кількість модулів у кожній підсистемі – вона не повинна перевищувати 30 модулів. Дане обмеження обумовлено обмеженістю обчислювальної потужності сучасної мікропроцесорної техніки під час рішення класу NP-повних задач.

Ключові слова: інформаційна система; функціональна стійкість; зовнішні та внутрішні дестабілізуючі фактори; моніторинг; діагностування; тестовий контроль; електростанція; ймовірність.