Method of Ensuring the Functional Stability of the Information System based on Detection of Intrusions and Reconfiguration of Virtual Networks

Iryna Zamrii[1], Viktor Vyshnivskyi[1], and Valentyn Sobchuk[2]

[1] *State University of Information and Communication Technologies, 7 Solomianska str., Kyiv, 03110, Ukraine*
[2] *Taras Shevchenko National University of Kyiv, 64 Volodymyrska str., Kyiv, 01033, Ukraine*

Abstract
The functioning of the information system takes place in conditions of constant interaction with the external environment under the influence of various destabilizing factors. Informational conflicts that provide information about bilateral interaction and have a destructive effect on the elements of the opposite party deserve special attention, which allows for obtaining, storing, and processing information necessary to achieve the goals of the entire system and even counteract the processes that have arisen under the influence of an informational conflict. Destabilizing factors and conflicts in the system lead to failures in the functional processes of the information system. Prevention of these effects occurs by ensuring the functional stability of the information system, that is, the ability of the system to preserve or restore certain system functions during the action of destabilizing factors. The article develops a method for ensuring the functional stability of the information system using software-defined wide area networks, which is aimed at solving the problem of increasing the stability and security of the information system against violations based on the detection of intrusions and the reconfiguration of virtual networks in virtual cloud environments.

Keywords
Information system, software-defined wide area networks, reconfiguration, functional stability, security.

1. Introduction

In today's environment, companies are forced to transform all areas of their activities, using digital technologies to increase efficiency, speed of execution, and cost optimization [1, 2]. As a result of these changes, the traditional approach of centralizing applications, network centers, and security services no longer guarantees the performance of these applications [3–6].

Another factor that has affected the provision of network and security services is the location of workers and users. Traditionally, users worked from a central office or branch office from where security and network services could be effectively delivered. But for now, users need to be able to access apps regardless of location. This means that enhanced security services must now be provided in all locations.

As technology evolves, it makes sense to consider enabling users to securely access applications from anywhere, whether they are hosted in the cloud or on a private host, locally or remotely, ensuring consistent security and transparency for users regardless of access method, as well as protecting the company's digital assets [7]. For this purpose, the integration of technologies necessary to provide users with secure access to data and programs regardless of location is increasingly used [8–11]. At the same time, an important component remains the maintenance of the normal functioning of the information system in conditions of constant destabilizing factors. The essence of this is to adopt countermeasures against various destabilizing factors [12], adapt functional algorithms to new conditions, organize functional restoration or ensure continued functioning in conditions of

CPITS-2024: Cybersecurity Providing in Information and Telecommunication Systems, February 28, 2024, Kyiv, Ukraine
EMAIL: irinafraktal@gmail.com (I. Zamrii); vish_vv@ukr.net (V. Vyshnivskyi); v.v.sobchuk@gmail.com (V. Sobchuk)
ORCID: 0000-0001-5681-1871 (I. Zamrii); 0000-0003-1923-4344 (V. Vyshnivskyi); 0000-0002-4002-8206 (V. Sobchuk)

system failures, perform analysis and reliability assessment, and, based on these data, assess the stability of the information system [13–15].

Analysis and assessment of stability allow timely support and restoration of the main functions of the system in the required amount and even allow for the influence of the external environment as a result of the effects of destabilizing factors and changes in algorithms, operating conditions, and system structure [16, 17].

The development of approaches to solving the problem of synthesizing functionally stable systems is a complex process. One of the methods is the formation of a system of rules for effective management of the functional stability of the system [18–20], and its specific implementation, in particular, for solving optimization problems. That is, in the problem of system stability synthesis, the development of principles and methods of ensuring the functional stability of the system is carried out to solve the problem of its improvement.

The analysis of recent studies shows that one of the methods of increasing security and stability is the reconfiguration of the information system network. The article [21] discusses the problem of dependent reconfiguration of NFV. A distributed approach is proposed to guarantee consistency during dependent reconfiguration. This approach consists of a distributed multi-domain model that establishes the interaction between federation objects and a causal-coherent distributed orchestration algorithm based on this model.

The paper [22] aims to enhance the reliability and quality of service of power smart grids by searching for and applying reconfiguration-oriented solutions. A novel definition of recovery performance is provided in terms of automatic recoverability and unavailability rates.

The paper [23] presents a performance optimization algorithm for controller reconfiguration in fault-tolerant distributed model predictive control for large-scale systems.

The article [24] exploits the potential benefits of a blockchain system integrated with a software-defined network. A new cluster-structured routing protocol for IoT networks using blockchain-based architecture for SDN controllers is proposed. This helped solve performance and security issues.

An intelligent decision-making framework is necessary to ensure that the system as a whole survives external failures and attacks through autonomous reconfiguration. As a result, research [25] proposed an intelligent decision-making model supported by edge computing to address the problem of real-time failures and attacks.

However, there is a need to develop a performance, configuration, and security management apparatus to effectively use the information and hardware resources of the information system. For this purpose, a methodology should be developed to ensure the effectiveness of the functioning of the information system from the point of view of functional stability.

2. The Method of Increasing the Functional Stability of the Information System

Consider the objective function of the top of the information system (IS) hierarchy graph:

$$\mathcal{Y}_{q_i} \to max, \qquad (1)$$

where for each vertex that continues to function without failure, the IS continues to function at full capacity:

$$\mathcal{Y}^{*}_{q_i} = \sum_{f=1}^{N_f} \sum_{j=1}^{a_f} \sum_{i=1}^{N_{q_f}} \mathcal{N} \left|\mathcal{M}_{\Phi}(a_{fj}, q_{fi})\right| \mathcal{Y}(a_{fj}) k, \qquad (2)$$

$$k = \begin{cases} 1, \text{ if } q_{fi} \text{ executes on } q_i, \\ 0, \text{ otherwise,} \end{cases} \qquad (3)$$

$$\mathcal{Y}(a_{fj}) = \sum_{i} q_i \mathcal{Y}_{q_i}, \qquad (4)$$

where $\mathcal{Y}(a_{fj})$ is the general solution of the optimal scenario under the influence of destabilizing factors, which is the sum of the products of vertices q_i associated with an abnormal situation for solving the problem a_{fj} and the scenario of making the best decision for solving the given problem $\mathcal{Y}_{q_i}$, $\mathcal{M}_{\Phi}(a_{fj}, q_{fi})$ is the matrix of functioning of vertices and problems of the IS hierarchy graph.

If the set of executable functions is a constant for the objective function, then

$$\mathcal{Y}_{q_i} = \mathcal{Y}^{*}_{q_i}. \qquad (5)$$

If some vertex is unable to withstand an abnormal situation and continue to function in the same mode, the system imposes a constraint $Q_0\,(a_{fj})$ for failure to perform assigned tasks on the vertex in the form of a function that acquires negative values. The controller of the system, depending on how critical the failure of the vertex to complete or partially perform the assigned tasks, adjusts the restrictions on it.

The method of forming restrictions is proposed as follows: $Q_0^* \left(a_{fj}\right)$ is a restriction in the case of a partial loss of productivity, that is, of the assigned functions, and $\alpha \times Q_1^* \left(a_{fj}\right)$ is a restriction in the case of a proportional loss of productivity α. Then

$$Q_0 \left(a_{fj}\right) = Q_0^* \left(a_{fj}\right) + \alpha \times Q_1^* \left(a_{fj}\right). \qquad (6)$$

If the functions perform tasks with different quality, then, accordingly, they are predicted to win in this strategy $\mathcal{S}_0(a_{fj})$, that is, in the scenario of making the best decision for a_{fj}.

It is obvious that if the function consumes resources, then it ensures the fulfillment of tasks, such as data transfer, management of processing in distributed databases, etc. That is, the more the function consumes resources, the better the result. And the size of the restriction will depend on the quality of the performance of each of the tasks a_{fj}.

Thus, the objective function for the top-level vertex of the IS network hierarchy will be to maximize the wins in each of the strategies and minimize the constraints:

$$\mathcal{F}(a_{fj}) = \sum a_{fj} \Big(\big(\mathcal{S}_0(a_{fj}) - \mathcal{S}_m(a_{fj})\big) \times \delta(a_{fj}) + Q_0(a_{fj}) \Big) \to min, \qquad (7)$$

where $\delta\left(a_{fj}\right)$ is a weighting factor that allows the controller to determine the priority of the performed functions in the system.

Consider the case in which the IS is unable to perform the amount of tasks and functions that arose as a result of extraordinary situations. In this case, there is a possibility that there are both server problems and problems arising as a result of user actions, so restrictions can be imposed on both components.

In this regard, within the framework of the proposed methodology, algorithms have been developed both for users and for problems related to equipment, and their execution does not necessarily have to be simultaneous.

For the server algorithm, we denote the period of its execution by $\mathcal{T}_c$ and impose the following conditions:

$$\mathcal{T}_c = \mathcal{t}_c + k\mathcal{T}_k, \qquad (8)$$

where $\mathcal{t}_c$ is the execution time of the task decision, k is the number of times the user algorithm is executed during the period $\mathcal{T}_c$.

$\mathcal{T}_k$ is the execution period of the user algorithm.

The periods and execution times of the specified algorithms are shown in **Error! Reference source not found.**.

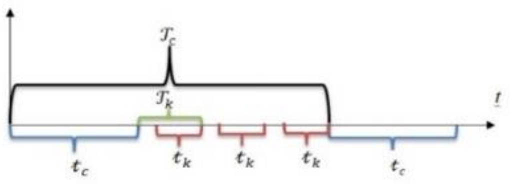


Figure 1: Periods and execution time of algorithms

The values of $\mathcal{T}_c$, $\mathcal{T}_k$, $\mathcal{t}_c$, k determine the order of application of algorithms in the methodology.

The developed technique consists of the following steps:

1. Definition of source data.
2. Using the Prisma Access infrastructure to bypass failed connections, including dynamic routing using BGP and information about new IP address subnets on the user connection side of the Prisma Access infrastructure to bypass failed connections when multiple routes exist between the client network and Prisma Access.
3. Checking whether all tasks a_{fj} have been completed. If not all a_{fj} are completed, then we proceed to the next step of the method.
4. Determination of $\mathcal{T}_c$, $\mathcal{T}_k$, $\mathcal{t}_c$, k using the IS controller reading the values of these parameters.
5. For each function f, the following are determined: $\mathcal{S}_0(a_{fj})$, $\delta\left(a_{fj}\right)$, $Q_0\left(a_{fj}\right)$ and methods are set.
6. $\mathcal{T}_c$, $\mathcal{T}_k$, $\mathcal{t}_c$, k are defined. The order of application and execution time of the methods is set.
7. Selection of the α value of the partial productivity of the IS.
8. Implementation of method $\mathcal{B}_1$.
9. Implementation of method $\mathcal{B}_2$.
10. Evaluation of the level of functional stability of the IS at the current moment using the SD-WAN cloud controller.
11. Determination of the need for correction of $\mathcal{T}_c$, $\mathcal{T}_k$, $\mathcal{t}_c$, k and assessment of the level of functional stability of IS.

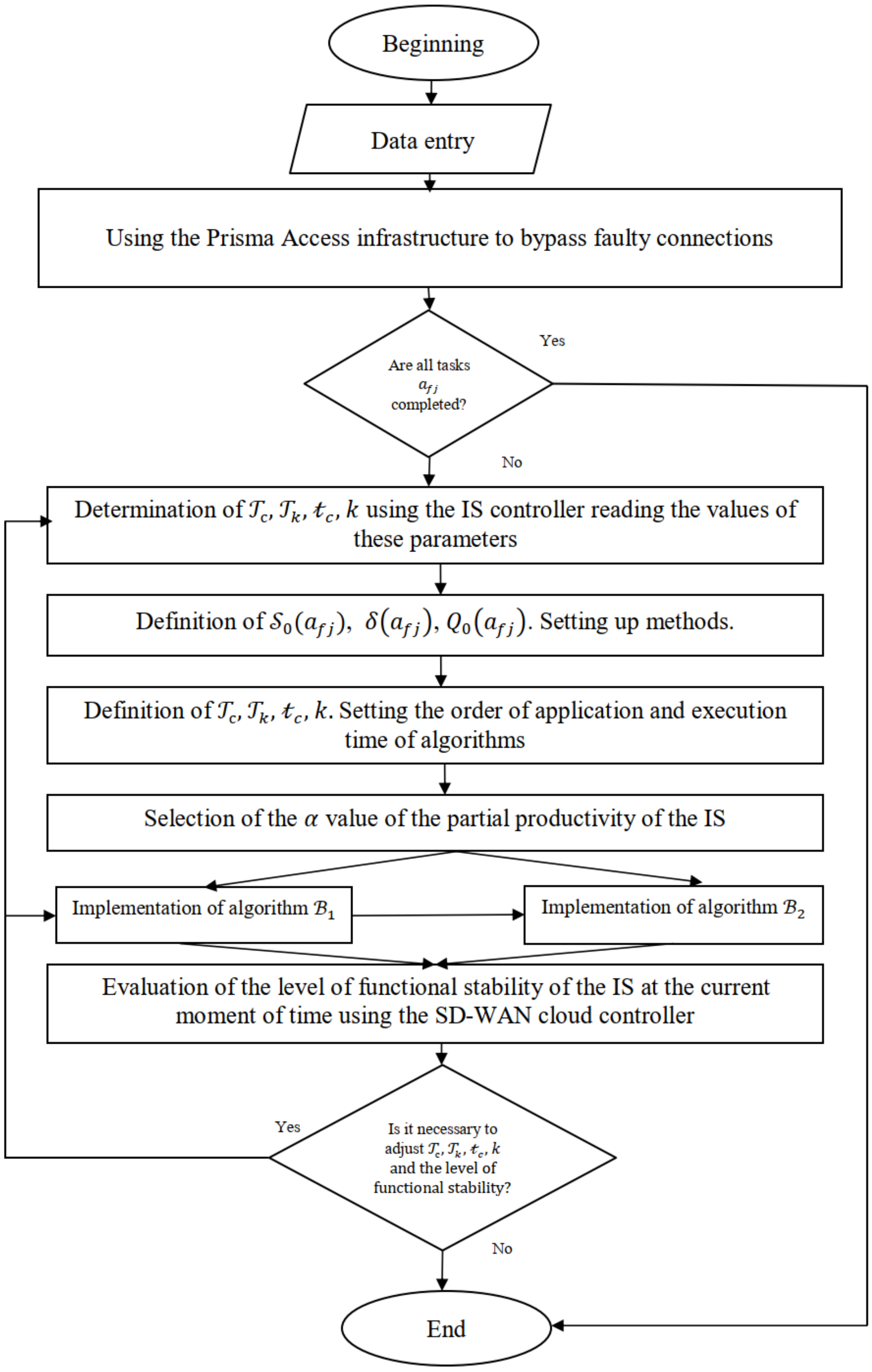

Beginning
Data entry
Using the Prisma Access infrastructure to bypass faulty connections
Are all tasks a_{fj} completed?
Yes
No
Determination of $\mathcal{T}_c$, $\mathcal{T}_k$, t_c, k using the IS controller reading the values of these parameters
Definition of $\mathcal{S}_0(a_{fj})$, $\delta(a_{fj})$, $Q_0(a_{fj})$. Setting up methods.
Definition of $\mathcal{T}_c$, $\mathcal{T}_k$, t_c, k. Setting the order of application and execution time of algorithms
Selection of the α value of the partial productivity of the IS
Implementation of algorithm $\mathcal{B}_1$
Implementation of algorithm $\mathcal{B}_2$
Evaluation of the level of functional stability of the IS at the current moment of time using the SD-WAN cloud controller
Is it necessary to adjust $\mathcal{T}_c$, $\mathcal{T}_k$, t_c, k and the level of functional stability?
Yes
No
End

Figure 2: Block diagram of the method of increasing the functional stability of the information system

In this methodology, $\mathcal{B}_1$ is an algorithm for the functional reconfiguration of the top of the graph of the top level of the IS network hierarchy, and $\mathcal{B}_2$ is an algorithm for the functional reconfiguration of the hierarchical IS network in real-time.

Visualization and order of interaction of the steps are presented in Fig. 2.

3. Algorithm of Functional Reconfiguration of the Top of the Graph of the Upper Level of the Hierarchy of the Information System Network

The selection of methods and mathematical apparatus for ensuring the functioning of the vertices of the graph of the hierarchical configuration of the network of the information system involves:

- Each of the functional levels must be able to make changes to the configuration and structural connections and take into account the possibility of breaking the connection with the top of the higher level of the hierarchy.
- The ability to quickly make changes in real time.
- Possibility of scaling.
- Possibility of use in dynamic models.

The need to ensure the specified requirements determines the possibility of applying the model of self-organizing systems in crises and the method of operational management of the theory of active systems.

According to the theory of active systems [26], a parameter called the incentive fund is introduced, the task of which is to be distributed among the vertices of the network. The optimal solution is to distribute the parameter between performers who have not yet performed their function. This distribution mechanism is centralized, which creates certain difficulties for the survivability of the system, in addition, even in distributed systems there are problems with the operation and distribution of this parameter.

The possibility of applying the distribution of this parameter due to the anticipatory self-monitoring approach solves the problem of timely response to failures in the vertices of the network graph and edges, that is, the connections associated with these vertices.

That is, during the performance of the functions assigned to the top, its behavior should be structured in such a way as to minimize restrictions (penalties in game theory [26]) and maximize decision-making strategies (incentives), due to which the system's efficiency increases. In addition, this approach is applicable during the functioning of the system in real-time.

Let's define tasks and connections between them in the hierarchical configuration of the information system network for algorithm $\mathcal{B}_1$.

The algorithm is designed to provide three types of functional restructuring of the information system network, namely:

- γ_1 is a functional rearrangement that will allow saving the set of all performed functions in the IC.
- γ_2 is a functional restructuring, which does not take into account the possible decrease in the quality of performance of specified functions under the influence of such restructuring.
- γ_* is a functional rearrangement that is a union of γ_1 and γ_2.

Since for the functioning of the IS in different periods, sets of different tasks and functions that have different levels of productivity can be performed, but despite everything, none of the functions can be neglected, the $\mathcal{B}_1$ algorithm is based on the priority execution of the functional rearrangement γ_1 and only under the condition the impossibility of ensuring the functioning of the system without losing the quality of part of the functions, the transition to functional restructuring γ_*.

The steps of performing the algorithm $\mathcal{B}_1$ include:

1. Entering input data, which includes the collection and processing of information about the vertices and the configuration of the network of the IS, namely: updating the data $\mathcal{D}(a_{fj},\mathcal{K})$ and $\mathcal{D}_c(a_{fj},\mathcal{K})$ by each of the vertices for tasks a_{fj}; update of current data from the network controller of the IS about the values of $\mathcal{T}_c$, $\mathscr{t}_c$.

2. Determination of the number of tasks performed by the vertex according to the formula:

$$\mathcal{R} = \sum_{f=1}^{N_f}\sum_{j=1}^{a_f}\sum_{i=1}^{N_{q_f}} \mathcal{N}\,|\mathcal{M}_\Phi(a_{fj}, q_{fi})|\mathcal{Y}(a_{fj})\mathscr{k} \qquad (9)$$

to calculate the complexity of decision-making in the performance of tasks to ensure the functioning of the IS network and the algorithm for further actions.

3. Determination of the tasks performed by the vertex at the current time. If the vertex does not perform tasks, then the method is not applied to this vertex.

4. Analysis of configurations $\mathcal{M}_\Phi(a_{fj}, q_{fi})(t)$ at the current time and

comparison with the previous configuration $\mathcal{M}_{\Phi}(a_{fj}, q_{fi})(t-1)$. Analysis and comparison take place sequentially for each task at a given top of the hierarchy. If $\mathcal{M}_{\Phi}(a_{fj}, q_{fi})(t)$ and $\mathcal{M}_{\Phi}(a_{fj}, q_{fi})(t-1)$ are identical, i.e. have no differences, then a check of changes in the system state is started.

5. Analysis of changes in the state of the IS network. If there have been changes in the network of the information system, that is, an increase or decrease in the total maximum speed of information transmission through specific vertices, then it is necessary to implement the main part of the algorithm.

6. Application of the approach of a complete search in one step. To apply the best configuration for the top-level vertex of the hierarchy, it is necessary to perform a restructuring depending on each of the tasks. Then, the top of the network executes a set of tasks to implement the best configuration in terms of performance and survivability, while its selection for each of the tasks is made taking into account the maximum speed in the network and the amount of system resources required for the calculation.

6.1. Searching for all possible configurations with lower requirements for computing and channel resources. If there is no simplification of the configuration, then the network is forced to stop performing part of the tasks.

6.2. Determination of the time t_{step} for the execution of one step of the approach with a review of all configuration options and determination of the time t_c of the execution of the solution of the task by the server algorithm using two-dimensional packaging with a full review of all options using the formula:

$$t_{\mathrm{c}}^{compute} = t_{step}\mathcal{J}_{a_{fj}}\mathcal{K}_{a_{fj}} + t_{realization}, \qquad (10)$$

where $t_{realization}$ is the time to apply the current configuration into action; $\mathcal{K}_{a_{fj}}$ is the configuration for the task a_{fj}; $\mathcal{J}_{a_{fj}}$ is a condition of the correctness of the configuration, which ensures the execution of at least one of the functions of the task a_{fj}. The time $t_{\mathrm{c}}^{compute}$ for different vertices in the network may be different, but $t_{\mathrm{c}}^{compute} \leq t_{\mathrm{c}}$.

7. Calculation of $t_{\mathrm{c}}^{compute}$ according to equality (10).

8. Determining whether a functional reconstruction of γ_1 is possible for the IS network, based on $\mathcal{S}_0$ is the maximum possible assessment of the quality of execution of a_{fj} in the configuration $\mathcal{K}_{a_{fj}}$, and at this stage of execution, the most important thing is to maximize the total value of these estimates:

$$\mathcal{S}_c(q_{fi}) = \sum_{f=1}^{N_f}\sum_{j=1}^{a_f}\left(\mathcal{S}_0\left(a_{fj}, \mathcal{K}_{a_{fj}}^0\right)\right)\mathcal{N}(|\mathcal{M}_{\Phi}(a_{fj}, q_{fi})| \qquad (11)$$

where $\mathcal{S}_0$ is a non-decreasing function of the best possible value of the assessment of the quality of the task a_{fj} with the configuration $\mathcal{K}_{a_{fj}}^0$, $\mathcal{K}_{a_{fj}}^0 = \overline{1, a_{fj}}$.

All configurations have a number that depends on the growth of the maximum possible assessment of the quality of the task $\mathcal{S}_0$ and the following conditions are imposed on them:

$$\begin{cases} \mathcal{K}_{a_{fj}}^n \geq \mathcal{K}_{a_{fj}}^{n-1}, \\ \mathcal{S}_0\left(a_{fj}, \mathcal{K}_{a_{fj}}^n\right) \geq \mathcal{S}_0\left(a_{fj}, \mathcal{K}_{a_{fj}}^{n-1}\right), \end{cases} \qquad (12)$$

where $\mathcal{K}_{a_{fj}}^{n-1} = \overline{1, a_{fj}}$, $\mathcal{K}_{a_{fj}}^n = \overline{1, a_{fj}}$.

In addition, depending on the task, network and computing resources may vary.

A necessary condition for the performance of the assigned functions is their performance in the full scope of the task a_{fj} by the vertex q_{fi}, i.e.:

$$\begin{cases} R_i \geq \sum_{f=1}^{N_f}\sum_{j=1}^{a_f}(\mathcal{N}(|\mathcal{M}_{\Phi}(a_{fj}, q_{fi})|\ )^2 \times \mathcal{D}\left(a_{fj}, t_{realization, q_{fi}}, 1\right), \\ u_i(q_{fi}) \geq \sum_{f=1}^{N_f}\sum_{j=1}^{a_f}(\mathcal{M}_{\Phi}(|\mathcal{K}(a_{fj}, q_{fi})|\ )^2 \times \mathcal{D}(a_{fj}, t_{realization, q_{fi}}, 1) + \\ \qquad + \sum_{\mathcal{M}_{\Phi}(a_{fj}, q_{fi}) = q_{fi}} \mathcal{D}_c(a_{fj}, t_{realization, q_{fi}}, 1). \end{cases} \qquad (13)$$

The necessary values are calculated at step 6 of the algorithm.

The algorithm can be continued only if the necessary conditions are met. At this step, a complete review of configuration options and checking them for the possibility of implementation takes place in one step.

9. Determination of the method of choosing a configuration by comparing calculated and input data. Namely, when the inequality $t_c^{compute} - t_{step} > t_c$ is fulfilled, a complete enumeration of all configurations is impossible, and therefore a genetic method is performed, which ensures that the enumeration of configurations is interrupted at an arbitrary step and provides a result no worse than the current one.

10. Solving during the time $t_{step}\mathcal{J}_{a_{fj}}\mathcal{K}_{a_{fj}}$ the $\mathcal{W}_{max}$ the problem of maximizing the total benefit of choosing configurations without exceeding the maximum permissible loss of

functions. For this, a complete enumeration of all possible configuration options is used, and the result of the decision is a vector of configurations.

11. Setting the initial time $t_{gm} = t$ for the genetic method of sorting configurations $\vec{\mathcal{K}}$.

12. We set the required number of steps r_{gm} for the counter in the execution of the genetic method, after which the cycle is reset.

13. Execution of one step of the $\mathcal{W}_{max}$ problem by the genetic algorithm for finding the vector $\vec{\mathcal{K}}$ for which inequalities (13) hold.

14. We set the value $r_{gm} + 1$ for the counter.

15. We check the restrictions:

$$\frac{(t_{gm} + 1)(t - t_{gm})}{t_{gm}} > t_c. \qquad (14)$$

16. We perform the description of the search results using the genetic algorithm and inequalities (13) by entering the parameter θ into the matrix $\mathcal{O}(a_{fj}, q_{fi}, \theta)$ such that: $\theta = 1$ for network restrictions; $\theta = 2$ for resource constraints. If there is a lack of resources, then the value in the matrix is 1, otherwise, it is 0.

17. We calculate the lack of resources for vertices located below in the hierarchy:

$$\left[\begin{array}{l} \mathcal{D}\left(a_{fj}, \mathcal{K}_{a_{fj}}^{a_{fj}-1}, 1\right) = 1, \\ \mathcal{D}\left(a_{fj}, \mathcal{K}_{a_{fj}}^{a_{fj}-1}, 1\right) = 2. \end{array}\right.$$

18. We set the zero configuration in the presence of a lack of resources in step 17: $\mathcal{S}_0(\mathcal{K}_{a_{fj}}^{a_{fj}-1}) = 0$.

19. We determine whether a functional reorganization of γ_1 is possible for the IS network similarly to step 8.

20. If the functional reconstruction of γ_1 for the information system network is not possible, then the possibility of functional reconstruction of γ_2 is determined. To do this, the configuration is sent to all vertices. Implementation of γ_2 occurs in steps 21–31.

21. Setting the initial values for the system function selection algorithm.

22. Setting the initial values for the current tasks sorting algorithm.

23. Setting the initial values for the algorithm for sorting the tops of the hierarchy for the current task and function.

24. We set the system function counter to increase by one.

25. We set the system task counter to increase by one.

26. Set the hierarchy level counter to decrease by one.

27. The verification of the possibility of performing functions by vertices is carried out.

28. A vertex that cannot perform a function is assigned a configuration of zero, and the value of consumption of computing resources is 0.

29. The resulting condition for the system function selection algorithm is determined.

30. The resulting condition for the task selection algorithm is defined.

31. The resulting condition is determined by the algorithm for sorting the tops of the hierarchy for the current task and function.

32. Configuration application block: in the block, not updated data is sent to the nodes.

The block diagram of the functional reconfiguration of the top of the graph of the top level of the information system network hierarchy is shown in Fig. 3

We will evaluate the correctness of this algorithm according to the following criteria: the possibility of obtaining a solution for a finite number of steps; stability according to input data; and stability in calculations.

To check the fulfillment of the criteria, we will perform the following steps:

1. We define the critical sections of the algorithm. These include:
   - Steps 13–14 for the $\mathcal{W}_{\max}$ problem.
   - Step 15, since it is critical to determine whether the genetic algorithm for the $\mathcal{W}_{\max}$ the problem has been resolved.
   - Steps 24–31 for three loop algorithms.
2. We determine what restrictions are imposed on critical areas.

The execution of steps 13–15 implies the fulfillment of the necessary condition (13).

Execution of step 15 is not possible without the condition $r_{\text{gm}} \neq 0$. For this, in step 12, the number of counter steps is set to $r_{\text{gm}} = 0$, and in step 14, $r_{\text{gm}} = 1$. In this case, the counter will iterate over values from $[1, +\infty)$ and ensure the condition $r_{\text{gm}} \neq 0$.

For steps 24-31, conditions $i = 0, j = f_f$, $f = \ell$ must be met.

3. We determine the changes in the postconditions for the initial conditions of the critical sections and add them to the system of criteria for the correctness of the algorithm.

The correctness of ensuring steps 13–15 determines infinity in the cyclic genetic algorithm. That is, if $t \to \infty$ then $r_{\text{gm}} \to \infty$ and

$\lim\limits_{t \to \infty} \frac{t - t_{gm}}{t_{gm}} = \text{const}$. Then, in the last equality, the limit goes to ∞, and the right-hand side of the equality remains a constant number. Thus, in the cyclic genetic algorithm, the number of steps necessarily remains a constant number at $t_c \neq \infty$.

For step 15, the correctness conditions are defined in steps 11 and 14.

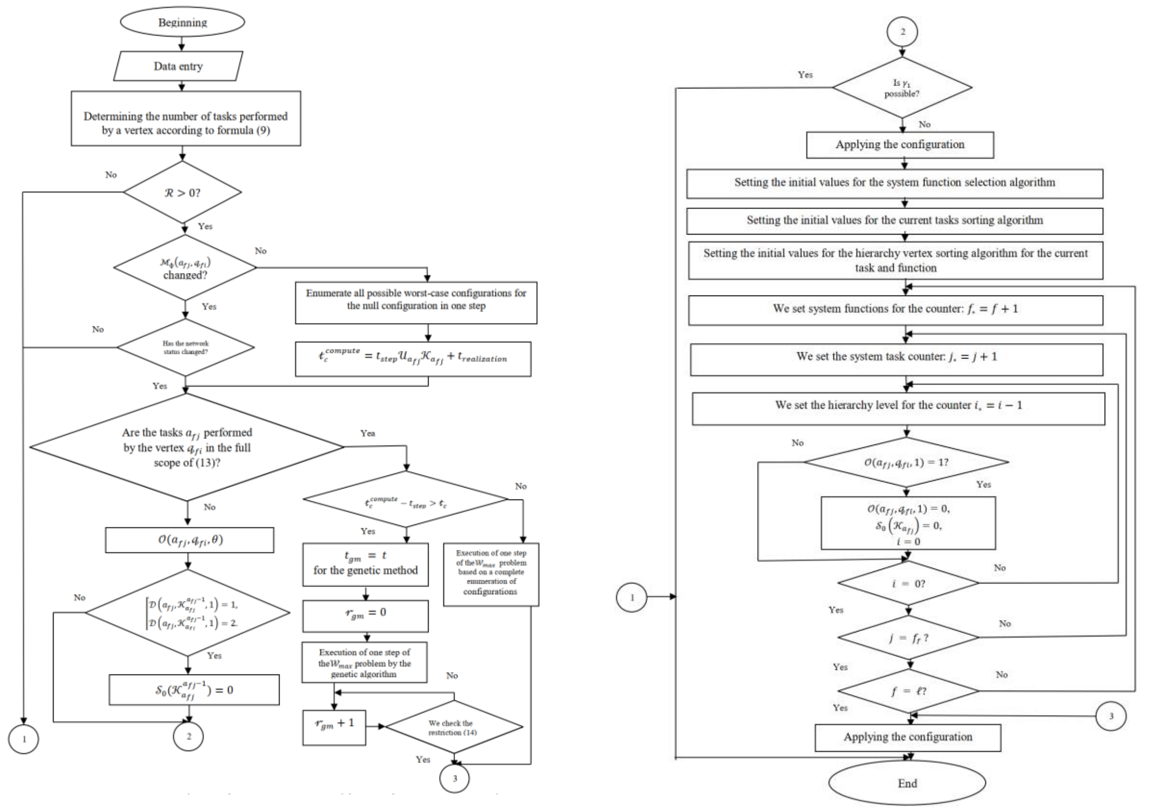
For steps 24–31, i is a natural number, so the number of configurations $\mathcal{K}_{a_{fj}} \geq 2$.

For step 30, the correct initial condition is a_f is a natural number for the postcondition, and for step 31 a_f is a natural number and $\ell > 0$.

Then the system of correctness conditions has the form:

$$\mathcal{J} = \begin{cases} a_{fj} \geq 0, \\ \mathcal{K}_{a_{fj}} \geq 2, \\ t_{gm} \geq 1, \\ \ell > 0. \end{cases} \tag{15}$$

Conditions (15) determine the execution of at least one function from a set of tasks. In addition, each of the tasks must have at least two levels of hierarchy to ensure a hierarchical configuration. The check is carried out in step 6.2.



Beginning
Data entry
Determining the number of tasks performed by a vertex according to formula (9)
No
$\mathcal{R} > 0$?
Yes
$\mathcal{M}_\Phi(a_{fj}, q_{fi})$ changed?
No
Enumerate all possible worst-case configurations for the null configuration in one step
Yes
No
Has the network status changed?
$t_c^{compute} = t_{step}\mathcal{U}_{a_{fj}}\mathcal{K}_{a_{fj}} + t_{realization}$
Yes
Are the tasks a_{fj} performed by the vertex q_{fi} in the full scope of (13)?
Yea
$t_c^{compute} - t_{step} > t_c$
No
No
$\mathcal{O}(a_{fj}, q_{fi}, \theta)$
Yes
$t_{gm} = t$ for the genetic method
Execution of one step of the W_{max} problem based on a complete enumeration of configurations
No
$\mathcal{D}(a_{fj}, \mathcal{K}_{a_{fj}}^{a_{fj}-1}, 1) = 1$, $\mathcal{D}(a_{fj}, \mathcal{K}_{a_{fi}}^{a_{fj}-1}, 1) = 2$.
$r_{gm} = 0$
Execution of one step of the W_{max} problem by the genetic algorithm
Yes
No
$\mathcal{S}_0(\mathcal{K}_{a_{fj}}^{a_{fj}-1}) = 0$
$r_{gm} + 1$
We check the restriction (14)
1
2
Yes
3
2
Yes
Is γ_1 possible?
No
Applying the configuration
Setting the initial values for the system function selection algorithm
Setting the initial values for the current tasks sorting algorithm
Setting the initial values for the hierarchy vertex sorting algorithm for the current task and function
We set system functions for the counter: $f_* = f + 1$
We set the system task counter: $j_* = j + 1$
We set the hierarchy level for the counter $i_* = i - 1$
No
$\mathcal{O}(a_{fj}, q_{fi}, 1) = 1$?
Yes
$\mathcal{O}(a_{fj}, q_{fi}, 1) = 0$, $\mathcal{S}_0(\mathcal{K}_{a_{fj}}) = 0$, $i = 0$
No
1
$i = 0$?
Yes
No
$j = f_f$?
Yes
No
$f = \ell$?
Yes
3
Applying the configuration
End

Figure 3: Block diagram of the functional reconfiguration of the top of the graph of the upper level of the hierarchy of the IS network

Thus, the method of functional reconfiguration of the top of the graph of the upper level of the IS network hierarchy is correct.

The problem of determining the best configuration for a_{fj} is NP-complete. A less optimistic option, which will ensure the performance of the system functions properly, is the tenth step of the method, in which a complete enumeration of configurations is carried out, the number of which can be defined as $\varphi(2k)$, the use of which for small values of k leads to a reduction in the execution time of the part of the algorithm, where k is defined as the product of the set of tasks of network functions by the set of vertices and the set of assumed configurations. If k acquires sufficiently large values, then the part of the method based on the genetic algorithm is executed. The complexity of the calculation depends on the number of transformations and the dimensions of the source data. The computational complexity of the algorithm due to the time limitation is inversely proportional to the computing power of the top of the information system network hierarchy graph.

4. Algorithm of Functional Reconfiguration of the Hierarchical Network of the Information System in Real Time

To ensure that external and internal influences on the IS will not lead to malfunctions, network reserves, computing reserves, and temporary reserves are provided, which are also called compensatory measures, and the mechanisms for their implementation are compensatory mechanisms [27]. In this method, in connection with the operation of IS in real-time, the temporary reserve is excluded.

In IS, the number and configuration of vertices and connections at the structural level can change, as can the number of functional elements and the list of functions they perform. The IS must quickly react to changes, therefore, during reconfiguration, the Q_{max} of vertices and connections is recalculated.

Let's denote the number of functions performed by the IS in the current state by $N = \sum_{i=1}^{m} N_{fi}$, where N_{fi} is the number of functions performed by the i-th functional element.

To be able to respond quickly, each vertex must have several solutions to choose the best, they provide a reserve of the necessary network and computing resources. In doing so, each vertex tries to perform the most "useful" function to try to maximize performance.

But according to the emerging various destabilizing factors, the vertices cannot always predict which of the solutions, in this case, will be better, therefore it is important to develop a method in which the adopted decision will be better for at least one of the vertices, and will not cause harm or damage to the rest. In addition, in real-time systems, it is necessary to ensure fast decision-making, and this can be ensured by pre-generated strategies.

It should be noted that with decentralized management, the system operates under conditions of uncertainty and therefore makes changes based on the data available to each node of the system. Therefore, it is necessary to check how it affects the vertices with which there is a direct connection and the system as a whole.

Let the decision regarding the presence of a reserve vertex be made at the level of the functional element μ.

For minor changes in the system not to lead to permanent reconfiguration, we will set the following requirements:

- Before starting the IS, the value of the minimum and maximum quality of performance of the functions is set, as well as the possibility of receiving a "reward", that is, a reserve of resources, in proportion to the quality of the performed function.
- "Bonus rewards" in the form of excess resources are also received for the performance of μ certain sets of functions.
- Restrictions are introduced for partial or complete loss of functionality.

These requirements are created in the function reconfiguration step for the corresponding μ and are determined by the following sequence of actions.

At the beginning, a list of functions up to μ is received. Combinations of functions by individual elements are added to this list in case of additional evaluations. This is followed by assigning a score to each feature based on priority to encourage support for existing functionality. The value of the extra point is small enough to prevent the system from making changes to the list based on available resources, performance, or other reasons.

It is necessary to introduce mechanisms for collecting information about the network and channel resources of μ vertices, synchronizing the list of functions, as well as redistributing additional estimates between the vertices of the hierarchy graph.

The method of complete selection of configurations or genetic is used for the $\mathcal{W}_{\max}$ problem of maximizing the total benefit of choosing configurations without exceeding the maximum allowable loss of functions in the case when each of the vertices has both a computing and a network resource. The choice of the method is due to strict limitations on the time of execution of the functions.

The choice of the most suitable strategy is made to preserve the functionality, so each vertex checks the list of actions to ensure the performance of the set functions.

During the execution of a new cycle by the system, the vertices analyze the results of the changes made to the configuration and calculate new combinations of functions and their possible additional evaluation.

Therefore, the IS network is organized in such a way that allows μ and vertices to make decisions independently, refusing centralized management and ensuring efficient operation in conditions of functional degradation.

We will describe the steps of the algorithm of functional reconfiguration of the hierarchical network of the information system in real-time.

1. We fix the start time t_θ.
2. The vertex q_{fi} is determined for the implementation of the configuration, by selecting neighboring vertices and some remote vertices of the same functional level, which can perform the same functions and between which connections are formed. In addition, the number of neighboring vertices is inversely proportional to the number of remote ones.
3. The vertices selected in step 2 exchange the matrices of functioning $\mathcal{M}_\Phi(a_{fj}, q_{fi})$ and matrices of functional possibilities $\mathcal{M}_\text{M}(a_{fj}, q_{fi})$.
4. It is determined which of the functions placed on the vertex q_fi, it is capable of performing:
$$\mathcal{M}_\Phi(a_{fj}, q_{fi}) = \mathcal{M}_\Phi(a_{fj}, q_{fi}) \times \mathcal{M}_\text{M}(a_{fj}, q_{fi}). \quad (16)$$
5. The number of tasks for the vertex q_{fi} that it can perform is determined by the formula:
$$\mathcal{R} = \sum_{f=1}^{N_f}\sum_{j=1}^{a_f}\sum_{i=1}^{N_{q_f}} \mathcal{N}\,|\mathcal{M}_\Phi(a_{fj}, q_{fi})|\,k. \quad (17)$$
6. When $\mathcal{R} > 0$, the search for a solution for the functional reconstruction of vertices and their connections is initiated. If a solution is found, the search result is implemented in steps 7–9, otherwise, the results of calculations from the neighboring node are expected.
7. The reference point for performing the genetic method for the $\mathcal{W}_\text{max}$ problem is determined.
8. The solutions of the $\mathcal{W}_\text{max}$ the problem is supplemented by restrictions imposed on the choice of configuration and the volume of tasks to be performed. At the same time, the genetic method is used, since these constraints require an increase in computing resources and time to search for a solution, and the result is no worse than the initial one.
9. Checking the maximum possible time for the implementation of the solution:
$$\frac{t(m+1) - (mt_\theta + t_\theta^*)}{m} > \mathcal{T}_k. \quad (18)$$
10. Calculation results are exchanged.
11. The top with the highest additional score is determined. If there is more than one such vertex, then the vertex that initiated the reconfiguration is selected.
12. The reconfiguration process is in progress.
13. Updating information in the vertices of the zero level of the hierarchy for each of the changed tasks.

The block diagram of the described algorithm is shown in Fig. 4.

Let’s evaluate the correctness of the algorithm. Critically important steps in the real-time hierarchical network functional reconfiguration algorithm are the sixth through the eighth. In this case, the completion of the cycle is possible only when (19) is executed. That is, correctness is determined by the following conditions:
$$\mathcal{T}_k \neq \infty, m \neq 0, \quad (19)$$
where $\mathcal{T}_k$ is the period of functional reconfiguration, m is the number of counter steps in the genetic method.

Since the conditions are checked in step 9, but first for the genetic method, then (19) are performed regardless of the obtained result.

Algorithm complexity assessment. For a complete search of all possible configurations, the complexity of the solution is defined as $\varphi(2k)$, but it is rational to implement it only for small values of k. For large values of k, a genetic method is used, the complexity of which is limited by the number of operations. Thus, the complexity of the calculation for one step can be defined as $\varphi(k^2)$. In steps 2–3, the complexity of the method is $\varphi(k)$ for each of them, which in sum gives $\varphi(2k)$.

Then, with a complete search, the complexity is $\varphi(2k + 2k^2)$ and with the genetic method, it is $\varphi(2k + 2k^2)$.

The other steps do not significantly affect the computational complexity.

The accuracy of the algorithm depends on the time of determination of the decision by the genetic method.

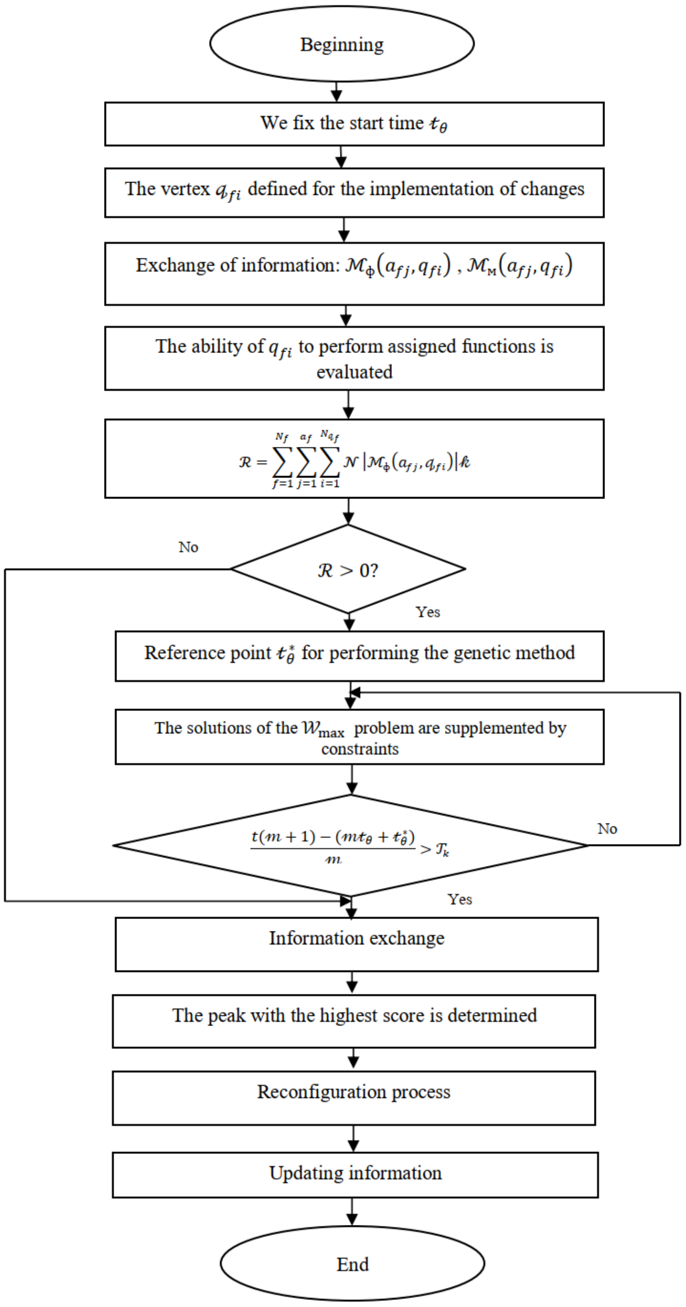

Beginning
We fix the start time t_θ
The vertex q_{fi} defined for the implementation of changes
Exchange of information: $\mathcal{M}_\text{ф}(a_{fj}, q_{fi})$, $\mathcal{M}_\text{м}(a_{fj}, q_{fi})$
The ability of q_{fi} to perform assigned functions is evaluated
$\mathcal{R} = \sum_{f=1}^{N_f} \sum_{j=1}^{a_f} \sum_{i=1}^{N_{q_f}} \mathcal{N} \left|\mathcal{M}_\text{ф}(a_{fj}, q_{fi})\right| k$
No
$\mathcal{R} > 0?$
Yes
Reference point t_θ^* for performing the genetic method
The solutions of the $\mathcal{W}_{\max}$ problem are supplemented by constraints
$\frac{t(m+1) - (mt_\theta + t_\theta^*)}{m} > \mathcal{T}_k$
No
Yes
Information exchange
The peak with the highest score is determined
Reconfiguration process
Updating information
End

Figure 4: Block diagram of the functional reconfiguration of the hierarchical network of the IS in real-time

5. Conclusions

The conducted analysis of the requirements for functionally stable information systems revealed the expediency of implementing software-defined wide area networks, which help ensure security, productivity, reliability, providing flexibility and manageability to network services.

A method of ensuring the functional stability of the information system using software-defined wide area networks has been developed, which differs from the existing ones in that it is based on the detection of destabilizing factors and the reconfiguration of the information system network into which the Prisma Access solution is integrated.

The use of this method helps ensure the functional stability of the information system by ensuring availability, integrity, confidentiality, and protection against unauthorized access, as well as preserving network bandwidth.

References

[1] F. Kipchuk, et al., Assessing Approaches of IT Infrastructure Audit, in: IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (2021). doi: 10.1109/picst54195.2021.9772181.

[2] V. Buriachok, V. Sokolov, P. Skladannyi, Security Rating Metrics for Distributed Wireless Systems, in: Workshop of the 8th International Conference on "Mathematics. Information Technologies. Education:" Modern Machine Learning Technologies and Data Science, vol. 2386 (2019) 222–233.

[3] P. Anakhov, et al., Protecting Objects of Critical Information Infrastructure from Wartime Cyber Attacks by Decentralizing the Telecommunications Network, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 3550 (2023) 240–245.

[4] H. Hulak, et al., Dynamic Model of Guarantee Capacity and Cyber Security Management in the Critical Automated System, in: 2nd International Conference on Conflict Management in Global Information Networks, vol. 3530 (2023) 102–111.

[5] V. Grechaninov, et al., Formation of Dependability and Cyber Protection Model in Information Systems of Situational Center, in: Workshop on Emerging Technology Trends on the Smart Industry and the Internet of Things, vol. 3149 (2022) 107–117.

[6] V. Grechaninov, et al., Decentralized Access Demarcation System Construction in Situational Center Network, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems II, vol. 3188, no. 2 (2022) 197–206.

[7] P. Johnson, et al., Digital Innovation and the Effects of Artificial Intelligence on Firms' Research and Development—Automation or Augmentation, Exploration or Exploitation? Technol. Forecast. Social Change 179 (2022) 121636. doi: 10.1016/j.techfore.2022.121636.

[8] K. Basu, A. Hamdullah, F. Ball, Architecture of a Cloud-based Fault-Tolerant Control Platform for improving the QoS of Social Multimedia Applications on SD-WAN, 13th International Conference on Communications (COMM) (2020) 495–500. doi: 10.1109/COMM48946.2020.9142038.

[9] G. Blokdyk, Software-Defined WAN SD-WAN A Clear and Concise Reference, 5STARCooks (2018).

[10] G. Blokdyk, SD-WAN A Complete Guide, 5STARCooks (2021).

[11] O. Lemeshko, et al., Research of Improved Traffic Engineering Fault-Tolerant Routing Mechanism in SD-WAN, IEEE 3rd Ukraine Conference on Electrical and Computer Engineering (UKRCON) (2021) 187–190.

[12] V. Oglih, H. Patoka, Formation of Information Security System Under Conditions of Uncertainty of Influence of Destabilization Factors on the Basis of Neurofacle Networks, Econom. Scope (2022) 76–81. doi: 10.32782/2224-6282/177-13.

[13] P. Zuiev, et al., Development of Complex Methodology of Processing Heterogeneous Data in Intelligent Decision Support Systems, Eastern-European J. Enterprise Technol. (2020) 14–23. doi: 10.15587/1729-4061.2020.208554.

[14] S. Yevseiev, et al., Synergy of Building Cybersecurity Systems (2021). doi: 10.15587/978-617-7319-31-2.

[15] D. Denyer, et al., Exploring Reliability in Information Systems Programmes, Int. J. Project Manag. (2011) 442–454. doi: 10.1016/j.ijproman.2011.02.002.

[16] W. Zheng, et al., Robust Stability Analysis and Feedback Control for Networked Control Systems with Additive Uncertainties and Signal Communication Delay Via Matrices Transformation Information Method, Inf. Sci. (2022) 258–286. doi: 10.1016/j.ins.2021.09.005.

[17] V. Zavgorodnii, et al., Methods and Models for Assessment of Reliability of Structural-Complex Systems, World Sci. (2018) 5–14. doi: 10.31435/rsglobal_ws/30112018/6227.

[18] O. Mashkov, et al., Application of the Theory of Functional Stability in the Problems of Covering Territories by Sensory Networks, Lecture Notes on Data Engineering and Communications Technologies 149 (2023) 266–285. doi: 10.1007/978-3-031-16203-9_16.

[19] V. Sobchuk, et al., Methodology for Building a Functionally Stable Intelligent Information System of a Manufacturing Enterprise, Bulletin of the Taras Shevchenko National University of Kyiv, Physics and Mathematics 4 (2021) 116–127. doi: 10.17721/1812-5409.2021/4.18.

[20] V. Sobchuk, I. Zamrii, S. Laptiev, Ensuring Functional Stability of Technological Processes as Cyberphysical Systems Using Neural Networks, Smart Technologies in Urban Engineering, LNNS 536 (2023) 581–592. doi: 10.1007/978-3-031-20141-7_53.

[21] J. Castañeda, et al., VNF-Based Network Service Consistent Reconfiguration in Multi-Domain Federations: A Distributed Approach, J. Netw. Comput. Appl. (2021) 103226. doi: 10.1016/j.jnca.2021.103226.

[22] S. Meskina, et al., Reconfiguration-Based Methodology for Improving Recovery Performance of Faults in Smart Grids, Inf. Sci. (2018) 73–95. doi: 10.1016/j.ins.2018.04.010.

[23] A. Zakharov, et al., A Performance Optimization Algorithm for Controller Reconfiguration in Fault Tolerant Distributed Model Predictive Control, J. Process Control (2015) 56–69. doi: 10.1016/j.jprocont.2015.07.006.

[24] S. Latif, et al., AI-Empowered, Blockchain and SDN Integrated Security Architecture for IoT Network of Cyber Physical Systems, Comput. Commun. (2021) 274–283. doi: 10.1016/j.comcom.2021.09.029

[25] Y. Yuan, et al., Model Exploration of Grid Adjustment and Restoration Strategy Based on Intelligent Decision System, Int. J. Thermofluids (2024) 100580. doi: 10.1016/j.ijft.2024.100580.

[26] O. Dodonov, M. Kuznetsova, O. Horbachyk, Survivability of Complex Systems: Analysis and Modeling, 2nd ed., Polytechnic (2009).

[27] S. Gao, J. Wang, J. Zhang, Reliability Analysis of a Redundant Series System with Common Cause Failures and Delayed Vacation, Reliability Eng. Syst. Safety (2023),109467. doi: 10.1016/j.ress.2023.109467.