Computing, Telecommunication and Control, 2024, Vol. 17, No. 2, Pp. 24–37.
Информатика, телекоммуникации и управление. 2024. Том 17, № 2. С. 24–37.

Software of Computer, Telecommunications and Control Systems

Программное обеспечение вычислительных, телекоммуникационных и управляющих систем

Research article
DOI: https://doi.org/10.18721/JCSTCS.17203
UDC 004.052.3

ALGORITHM FOR MONITORING AND IMPROVING THE STABILITY OF THE IT INFRASTRUCTURE BASED ON AVAILABILITY AND RELIABILITY METRICS

D.A. Varlamov, I.V. Nikiforov ✉ , *S.M. Ustinov*

Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russian Federation

✉ igor.nikiforovv@gmail.com

Abstract. Most companies have their own IT infrastructure that consists of complex systems and services. The stability of systems and services is important for companies, as problems with them can lead to loss of resources and human time. Thus, it is important to analyze previous IT service outages, which aims to identify and adjust the most critical and vulnerable elements of the infrastructure that are prone to breakage or failure. *Research objective* is to develop a new algorithm for improving the stability of IT infrastructure of a company by analyzing and taking into account the statistics of previous services outages. As *a result*, a new algorithm is proposed to identify and fix problems in IT services before they lead to serious consequences and reduce the time to find the source of problem. The algorithm is based on two new metrics: availability and reliability, which distinctive feature is the consideration of statistics of previous failures and outages in the system. The architecture of a high-performance software tool that allows real-time monitoring and evaluation of IT services stability metrics is presented. The effectiveness of the proposed algorithm is demonstrated by implementing it in a software tool and observing the growth of stability indicators – availability and reliability – after the detection and elimination of a weak link in IT services. The use of the developed algorithm allowed to reduce the time during which the material and human resources of the company were idle by 25%. The practical significance of the presented algorithm was tested in one of the large industrial information technology companies with more than 10000 employees. Based on the information obtained with created software, it was possible to obtain recommendations for improving the stability of company's IT services.

Keywords: metrics, availability, reliability, stability, IT infrastructure, outage, monitoring

Citation: Varlamov D.A., Nikiforov I.V., Ustinov S.M. Algorithm for monitoring and improving the stability of the IT infrastructure based on availability and reliability metrics. Computing, Telecommunications and Control, 2024, Vol. 17, No. 2, Pp. 24–37. DOI: 10.18721/JCSTCS.17203

Научная статья
DOI: https://doi.org/10.18721/JCSTCS.17203
УДК 004.052.3

АЛГОРИТМ МОНИТОРИНГА И ПОВЫШЕНИЯ СТАБИЛЬНОСТИ ИНФОРМАЦИОННО-ТЕХНОЛОГИЧЕСКОЙ ИНФРАСТРУКТУРЫ НА ОСНОВЕ МЕТРИК ДОСТУПНОСТИ И НАДЕЖНОСТИ

Д.А. Варламов, И.В. Никифоров ✉ , *С.М. Устинов*

Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

✉ igor.nikiforovv@gmail.com

Аннотация. Большинство компаний имеют собственную информационно-технологическую инфраструктуру, состоящую из сложных систем и сервисов. Стабильность работы сервисов важна для компаний, так как проблемы с ними приводят к потерям ресурсов и человеческого времени. Поэтому важным является анализ предыдущих отключений сервисов, который направлен на выявление и налаживание уязвимых элементов инфраструктуры, подверженных поломке или отказу. *Цель исследования*: разработать алгоритм для повышения стабильности информационно-технологической инфраструктуры предприятия за счет анализа и учета статистики предыдущих отключений. *Результаты*: предложен новый алгоритм, позволяющий выявлять и устранять проблемы в информационно-технологических сервисах предприятия до того, как они приведут к серьезным последствиям, и сокращать время на поиск источника проблемы. Алгоритм основан на двух новых метриках: доступность и надежность, – отличительной особенностью которых является учет статистики предыдущих отключений. Представлена архитектура высокопроизводительного программного средства, позволяющего в режиме реального времени осуществлять мониторинг и оценку показателей стабильности сервисов. Демонстрируется эффективность предложенного алгоритма путем его реализации в программном средстве и наблюдения роста показателей стабильности – доступности и надежности – после обнаружения и устранения слабого звена в информационно-технологических сервисах. Использование разработанного алгоритма позволило на 25% сократить время, в течение которого материальные и человеческие ресурсы компании простаивали. *Практическая значимость*: представленный алгоритм применен на практике в одной из крупных промышленных информационно-технологических компаний с более чем 10000 сотрудников. На основе информации, полученной при помощи созданного программного средства, удалось получить рекомендации по повышению стабильности информационных сервисов компании.

Ключевые слова: метрики, доступность, надежность, стабильность, информационно-технологическая инфраструктура, отключение, мониторинг

Для цитирования: Varlamov D.A., Nikiforov I.V., Ustinov S.M. Algorithm for monitoring and improving the stability of the IT infrastructure based on availability and reliability metrics // Computing, Telecommunications and Control. 2024. Т. 17, № 2. С. 24–37. DOI: 10.18721/JCSTCS.17203

Introduction

Stability of internal services is important for all companies using information technology infrastructure. The stability of IT services affects the speed and quality of work of all employees of the company [1]. Any downtime of services entails costs and losses for the company. Analysis of previous IT service outages is designed to find the most critical and vulnerable infrastructure elements that are prone to breakage or failure to predict in advance which of the elements have a greater probability of failure [2]. This process allows you to solve problems in the infrastructure before they lead to huge resource and human time losses as it reduces the time to find the source of the problem. That is why it is urgent to create, improve and

automate algorithms to increase the stability of the IT infrastructure [3, 4]. The main components of the IT infrastructure that stand out in our work are software, hardware, and computer networks.

The article [5] proposes to assess the stability of the IT infrastructure based on user perception. The authors suggest using one metric to assess stability: availability. Various options for calculating this metric are offered, including those that take into account the severity of failures. In our work, we propose to use not one, but two metrics to assess the stability of the IT infrastructure: availability and reliability. We distinguish two classes of problems with the IT infrastructure: outage and degradation. Outage refers to a condition in which the service cannot perform its basic functionality. Degradation refers to a condition in which the service is able to perform its basic functions, but some of the additional functions cease to work correctly. We cannot use a single metric to evaluate a system that can be in one of three states: healthy, degrading or failing, because it is unclear to what extent degradation and to what extent outages will affect it. We need to have two metrics: one will only be affected by outages, and the second one will be affected by degradations and outages. Considering both classes of problems, the combination of the two metrics will give us a more informative assessment of IT infrastructure stability.

The goal of the work is to develop and implement in the software a new algorithm for improving the stability of IT infrastructure, a distinctive feature of which is the accounting of statistics of previous outages. To achieve this goal in the work, it is required to perform the tasks listed below.

1. To study existing solutions to improve IT infrastructure stability.
2. To study existing algorithms to calculating key indicators of IT infrastructure stability.
3. To propose a new algorithm for calculating key indicators of IT infrastructure stability, taking into account the state of degradation.
4. To propose the algorithm for improving stability of IT infrastructure. A distinctive feature of the algorithm is the accounting of statistics of previous outages.
5. Implement the proposed algorithm in a software tool.
6. Evaluate the effectiveness of the developed algorithm and the software that implements it.

Overview of solutions to improve stability

There are various algorithms for improving stability. For example, in the article [6] authors consider an algorithm to increase the stability of IT services in large companies with overlapping IT frameworks. To improve the quality of IT services, they propose to use a combination of reliability, assurance, responsiveness, empathy and tangibles to improve the quality of services. However, they focus on providing IT services as a service, and in our article, we focus specifically on IT infrastructure within the company.

The article [7] describes an algorithm for monitoring and measuring the quality for the Internet of Services (IoS) – an ecosystem in which online and offline services provided by many different service providers are intertwined. The authors developed and implemented an algorithm based on establishing a hierarchy of indices divided into three types: value, quality and capability (VQC). After dividing indices, service providers establish a dynamic hierarchy that defines the calculation relationship between indices, which helps to monitor the stability of the system in real time. This algorithm is effective when it comes to integrating services from different service providers. However, the company's IT infrastructure often has one or very few service providers, and in this case, the advantages of the VQC algorithm, tailored for IoS, are not fully used.

The article [8] contains various approaches to using machine learning to improve the security and stability of power systems. Machine learning allows you to quickly identify and detect problems related to cyberattacks, voltage instability, power quality disturbance, etc. The article discusses various algorithms to the implementation of machine learning: artificial neural networks (ANN), decision tree (DT), support vector machines (SVM). The same algorithms can be used not only to improve the stability of power systems, but also to apply them to IT systems. However, machine learning methods have a number of disadvantages: firstly, a large amount of data is needed for training the model. It may take

years to collect enough data for datasets from monitoring systems. Secondly, they have low estimation accuracy and often make mistakes, and therefore are poorly applicable in big IT systems where the price of mistake is very high.

In the article [9], the authors proposed a novel clustering-based algorithm Log3C, which allows you to promptly and precisely identify impactful system problems by utilizing log sequences and key performance indicators (KPIs) of the system. This algorithm was evaluated on real logs collected from the Microsoft online service system, and the results confirmed its effectiveness. However, implementing this approach on a scale of the whole IT infrastructure of a company may be accompanied by scaling problems. Collecting logs from all IT services and then analyzing all these logs can lead to high performance overhead and using this algorithm may not be so effective regarding the computing and storage resources that it needs.

In our work, we implement a solution based on the use of metrics to assess stability and search for vulnerable elements and weak links in the IT infrastructure, which is a distinctive feature of the proposed algorithm. Our algorithm distinguishes the degradation and outage states of the services. This helps us to calculate two more accurate metrics that shows us vulnerable elements of the IT infrastructure that are prone to problems.

Comparative analysis of existing algorithms to calculating stability metrics

A comparative analysis of existing algorithms for calculating IT infrastructure stability metrics needs to be conducted in order to identify their advantages and disadvantages.

Availability [10] is calculated by the formula

$$Availability = 100\% \times \frac{AST - DT}{AST}, \tag{1}$$

where AST is the agreed service time and DT is the sum of downtime. The advantages of this metric include ease of calculation and data collection. However, this metric has a disadvantage: it does not take into account the severity of failures. The severity of a failure means how much it affects the company's production process: slows it down or stops it completely.

Service Level Agreements (SLA) Compliance Ratio [11] is a compliance coefficient for SLA. It is calculated by the formula

$$SLA\ Compliance\ Ratio = \frac{N_{sla}}{N_{fail}}, \tag{2}$$

where N_{sla} is the number of outages, resolved in compliance with SLA, and N_{fail} is the number of outages. The advantage of this metric is that the resulting number has visual clarity for the client, and therefore inspires confidence. However, the disadvantage of using this metric is the need to document realistic requirements that can be implemented from the very beginning.

Mean Time Between Failures (MTBF) [12] is calculated by the formula

$$MTBF = \frac{T_{el} - T_{dt}}{N_{fail}}, \tag{3}$$

where T_{el} is the total elapsed time, T_{dt} is the total downtime and N_{fail} is the number of failures.

Mean Time To Repair (MTTR) [13] is calculated by the formula

$$MTTR = \frac{T_{main}}{N_{fail}}, \tag{4}$$

where T_{main} is the total maintenance time and N_{fail} is the number of failures.

The advantage of the MTBF and MTTR metrics is that they allow us to understand how well the service will be available in the context of various real-world conditions. However, their disadvantage is that they do not take into account the severity of failures.

First contact resolution rate (FCRR) [14] is calculated by the formula

$$FCRR = \frac{NL_1}{N_{fail}}, \tag{5}$$

where NL_1 is the number of incidents resolved by the first line support and N_{fail} is the number of failures. The advantage of this metric is its convincing economic justification. However, its disadvantage is that this metric depends more on the quality of IT infrastructure support than on its smooth operation.

Based on the comparative analysis given in Table 1, it can be concluded that all metrics have strengths indicated in the "Advantages" column, but none of them can be called optimal and universal, since each of them has their weaknesses indicated in the "Disadvantages" column. Therefore, it is proposed to introduce new metrics for the stability of the IT infrastructure.

Table 1

Comparison between different metrics for calculating stability of the IT-infrastructure

Metric	Formula	Advantages	Disadvantages	Range of possible values
Availability	$\frac{AST - DT}{AST} \times 100$	Ease of calculation and data collection	The severity of failures is not taken into account	From 0% to 100%
SLA Compliance Ratio	$\frac{N_{sla}}{N_{fail}}$	The resulting number has visual clarity for the client	Need to document realistic requirements that can be implemented from the very beginning	From 0 to 1
MTBF	$\frac{T_{el} - T_{dt}}{N_{fail}}$	Allows us to understand how well the service will be available in the context of various real-world conditions	The severity of failures is not taken into account	From 0 seconds to the entire measurement period
MTTR	$\frac{T_{main}}{N_{fail}}$	Allows us to understand how well the service will be available in the context of various real-world conditions	The severity of failures is not taken into account	From 0 seconds to the entire measurement period
FCRR	$\frac{NL_1}{N_{fail}}$	Convincing economic justification	Depends more on the quality of IT infrastructure support than on its smooth operation.	From 0 to 1

Improved stability metrics

To assess the effectiveness of the developed algorithm, it is proposed to introduce two stability metrics: availability and reliability.

Availability is calculated by the formula

$$Availability = 100\% \times \frac{AST - OT}{AST}, \quad (6)$$

where OT is the sum of outage time. Availability describes the extent to which the service can perform its basic functionality.

Reliability is calculated by the formula

$$Reliability = 100\% \times \frac{AST - OT - DT}{AST}, \quad (7)$$

where DT is the sum of degraded time. Reliability describes how much the service is able to perform all its functionality.

The advantage of these metrics is that when they are used together, the severity of failures can be taken into account. They can be calculated based on the data from existing monitoring systems. However, for their correct calculation, it is necessary to separate the outage (OT), and degradation (DT) state of a system.

The range of possible values of the proposed metrics is from 0% to 100%. At the same time, the "reliability" metric can never be greater than the "availability" metric. To reduce the number of calculations performed, the "reliability" metric can be calculated by the formula

$$Reliability = Availability - \frac{DT}{AST} \times 100\% \quad (8)$$

if the value of "availability" metric has already been calculated and the sum of degraded time is also known.

The availability and reliability metrics help us to evaluate the periods of IT-infrastructure stability. However, we need to focus on decreasing the time our infrastructure is instable, so we will also propose two metrics to evaluate instability time. They are called "Service absence" and "Service fragility" and calculated by formulas

$$Absence = 100\% - Availability \quad (9)$$

and

$$Fragility = 100\% - Reliability. \quad (10)$$

Service absence tells us about the percentage of time when the IT infrastructure was completely unavailable and could not be used, which caused resources loss. Service fragility tells us about the percentage of time when the IT infrastructure was either fully unavailable or some non-critical functions of it were broken which caused slowdown of company operations and also caused resources loss. The decreasing of these two metrics will help us understand how much IT downtime was reduced with our algorithm and eventually will give us an understanding of how many resources and production capacities we managed to save.

Proposed algorithm of failure analysis

We propose an algorithm for improving the IT infrastructure stability, and failure analysis is the distinctive feature of it. The "absence" and "fragility" metrics proposed in the paper will be calculated as

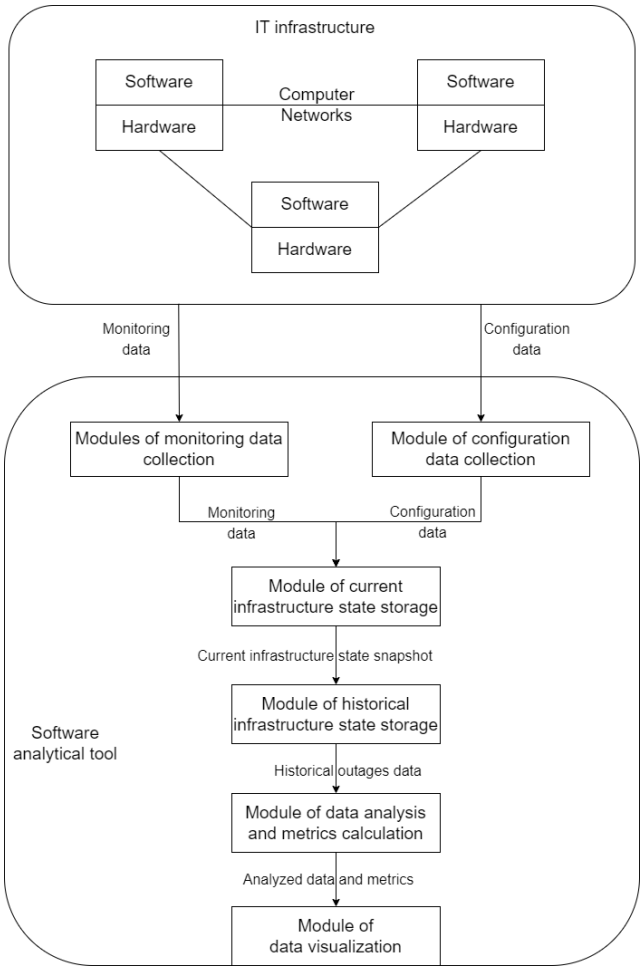

Fig. 1. Proposed algorithm of failure analysis

a part of this algorithm to help us detect the most vulnerable and unstable parts of IT infrastructure. This algorithm is presented in Fig. 1. The top of the figure shows an IT infrastructure consisting of three components: software, hardware and computer networks. The bottom of the figure shows the software analytical tool, which is an essential component of the algorithm. The software analytical tool consists of several modules that work independently.

The algorithm consists in collecting the data about IT infrastructure and passing it through all modules of the software analytical tool as in a pipeline. After the data completes all five stages of the pipeline, we get a visualized information about the most vulnerable components of the IT infrastructure. Each stage of a pipeline is presented by its own module and has its own distinctive features.

The first stage of the pipeline is divided in two parts that work in parallel. The first part is implemented in the modules of monitoring data collection. They are responsible for collecting monitoring data and logs from various distributed components of the IT infrastructure: software, hardware, and computer networks [15], which allows you to get a complete picture of the state of the IT infrastructure. The first part is presented by multiple modules, because each part of IT infrastructure has its own specifics in monitoring data, and a single module cannot collect all logs and metrics from all of IT infrastructure. The second part of the first stage is implemented in the module of configuration data collection. It is responsible for storing internal dependencies of IT infrastructure components on each other. In following stages, this data allows us to find and trace patterns in the behavior of outages, thus more accurately detect the causes of outages and most vulnerable components of IT infrastructure.

The second stage of the pipeline is implemented in the module of current infrastructure state storage. It is responsible for aggregating the data from multiple monitoring data collection modules and configuration data collection module. Such aggregation is implemented using the API [16] and helps us to get a complete understanding of the state of the entire IT infrastructure at a single moment in time. To speed up the module the API allows using "bulk update" request – a single request to update data on all components synchronously.

The third stage of the pipeline is implemented in the module of historical infrastructure state storage. It is responsible for storing the snapshots of connections between the components of the IT infrastructure and their states at each moment of time. This provides us a sufficiently large amount of data for calculating metrics and the ability to analyze previous outages that occurred during the previous years.

The fourth stage of the pipeline is implemented in the module of data analysis and metrics calculation. It is responsible for analyzing IT infrastructure outages, searching for dependencies between these outages and making recommendations to improve stability. This module also provides the calculation of the reliability and availability metrics proposed by us.

The fifth stage is the last stage of the pipeline. It is implemented in the module of data visualization. It is responsible for visualizing the aggregated and analyzed data on the state of IT infrastructure components, the connections between them and calculated metrics. This ensures the visibility of the calculated metrics for the end user, since it is clear where the calculated values came from, and which outages affected them. In addition, the observability of the IT infrastructure is ensured, which allows users to quickly find the source of outages and resolve incidents.

Features of the software implementation

The architecture described above was implemented using a stack of technologies. The software was created as subsystem inside a massive monitoring architecture of an enterprise company, so not all elements of the architecture were created from scratch. Fig. 2 shows the architecture of the implemented software and notes which software components already existed before our implementation, and which were implemented by us.

In particular, monitoring data collection modules were already implemented using Nagios, Zabbix and Solarwinds software products. The log collection module was implemented using the Elasticsearch software product. The IT infrastructure configuration storage module was implemented using the service-now software product.

In the course of the work, the module for storing data on the current health of the IT infrastructure was implemented in a software tool we created, called "Healthcheck DB". This software tool is written in the Golang language and uses a PostgreSQL database.

The module for storing historical data on the state of the IT infrastructure health was implemented in the course of work by configuring the Prometheus software along with VictoriaMetrics as remote database.

The module of data visualization was implemented in the course of work by configuring the Grafana software. At the same time, the module of data analysis and metrics calculation were implemented using queries from Grafana to Prometheus in Prometheus Query Language (PromQL), and the data visualization module was implemented using dashboards with visualizations [17] of various types in Grafana.

The deployment of all the modules we created was automated. Automation was created in the form of so-called "playbooks" in the configuration management tool Ansible, which allowed us to deploy the software we created on virtual machines with dedicated addresses on the network. At the same time, the software tools themselves work in isolated Docker containers.

In order to secure the collected data, backups of all used databases were created. Backups were implemented as automations in the Jenkins software. These automations are performed once a day, create a backup of each database and save it in the universal artifact repository manager Artifactory.

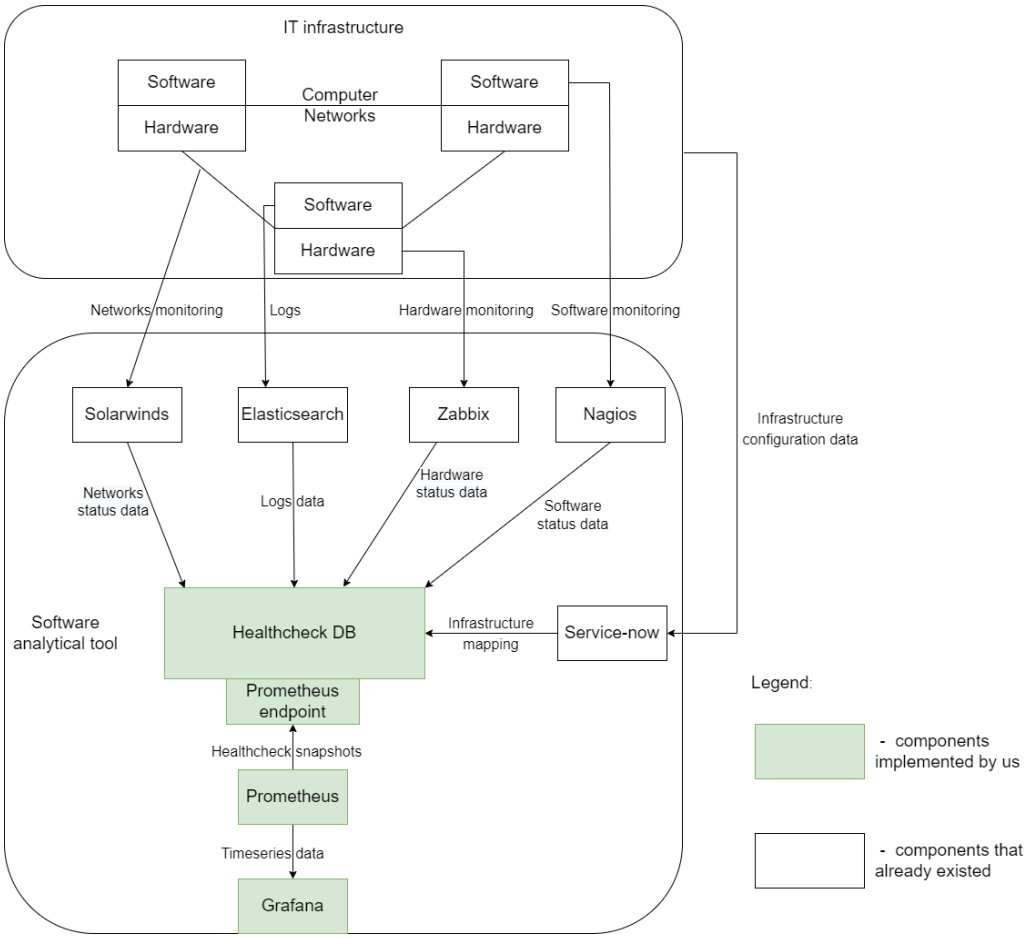

Fig. 2. Architecture of the implemented software

During the entire development process, the Jira software tool [18] was used for task management. To collaborate on code and version control, the GitHub software tool [19] was used. The Visual Studio Code software tool was used as a development environment for writing code. Confluence software was used to store the project documentation.

Description of the dashboards

As a result, several dashboards [17] were created, in order to search for and demonstrate elements of the IT infrastructure that are prone to outages.

Fig. 3 shows a dashboard that demonstrates reliability and availability metrics values for individual hosts of various services for different time intervals in the form of tables. The presented visualization allows you to quickly identify problem periods for each host of services and track the dynamics of changes in metrics in various time intervals: from the last 3 months to the last day. The dashboard also includes filters that allow you to select a list of necessary hosts of services, for example, those used by a certain production team. Thus, it is possible to assess the dynamics of stability of a certain segment of the IT infrastructure.

Fig. 4 shows a dashboard that demonstrates the values of reliability and availability metrics for services as a whole in the form of tables and graphs. The values in the graphs are discrete and are collected with 1 min frequency. The presented visualization allows you to quickly identify problematic periods of services and track the dynamics of changes in availability and reliability in various time intervals: from the last 3 months to the last day. Thus, it is possible to assess the dynamics of the stability of the entire IT infrastructure.

Availability metrics				
Host	Last 90 days	Last 30 days	Last 7 days	Last 24 hours
confluence1.example.com	100%	100%	100%	100%
confluence2.example.com	100%	100%	100%	100%
jira1.example.com	100%	100%	100%	100%
jira2.example.com	100%	100%	100%	100%
jenkins1.example.com	99.874%	99.582%	99.549%	99.814%
Reliability metrics				
Host	Last 90 days	Last 30 days	Last 7 days	Last 24 hours
confluence1.example.com	99.917%	100%	100%	100%
confluence2.example.com	100%	100%	100%	100%
jira1.example.com	99.653%	99.768%	99.356%	100%
jira2.example.com	100%	100%	100%	100%
jenkins1.example.com	97.944%	98.104%	98.156%	98.865%

Fig. 3. Dashboard with availability and reliability metrics for service hosts

Availability metrics				
Service	Last 90 days	Last 30 days	Last 7 days	Last 24 hours
Confluence Wiki	100%	100%	100%	100%
Jira	100%	100%	100%	100%
Jenkins	99.592%	99.145%	100%	100%
Kafka	100%	100%	100%	100%
Reliability metrics				
Service	Last 90 days	Last 30 days	Last 7 days	Last 24 hours
Confluence Wiki	97.152%	93.125%	100%	100%
Jira	99.667%	99.688%	100%	100%
Jenkins	95.430%	95.535%	97.256%	99.583%
Kafka	100%	100%	100%	100%

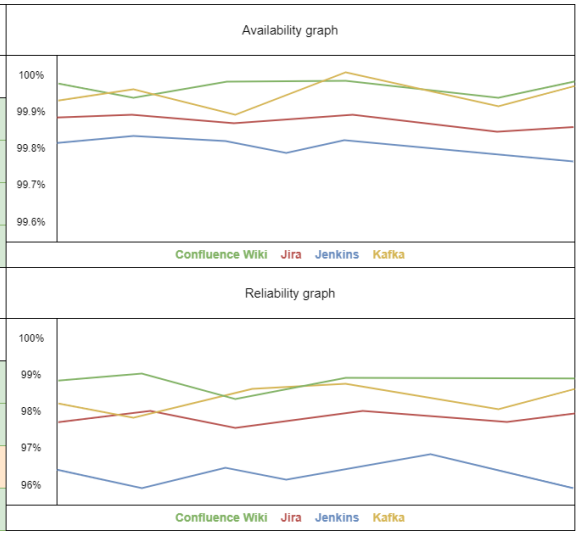

Fig. 4. Dashboard with availability and reliability metrics for services

Fig. 5 shows a dashboard that demonstrates the states of service hosts for a certain period (by default, for the last 24 hours) in the form of a heatmap [20]. On this dashboard, the red highlights the moments of time when service hosts were in the outage state, orange marks the state of degradation and green marks the “correct” state.

The presented visualization allows you to find general trends in service failures. For example, in Fig. 5 there is a simultaneous transition of several hosts of “Jenkins” service into a state of degradation at once, which suggests that these failures had a common cause.

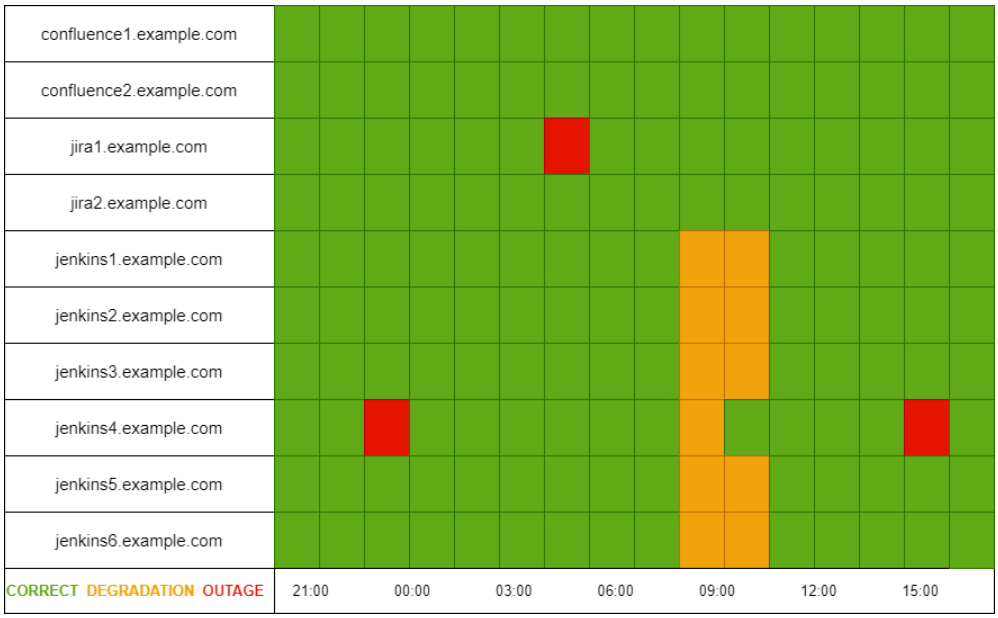

Fig. 5. Dashboard with service hosts states heatmap

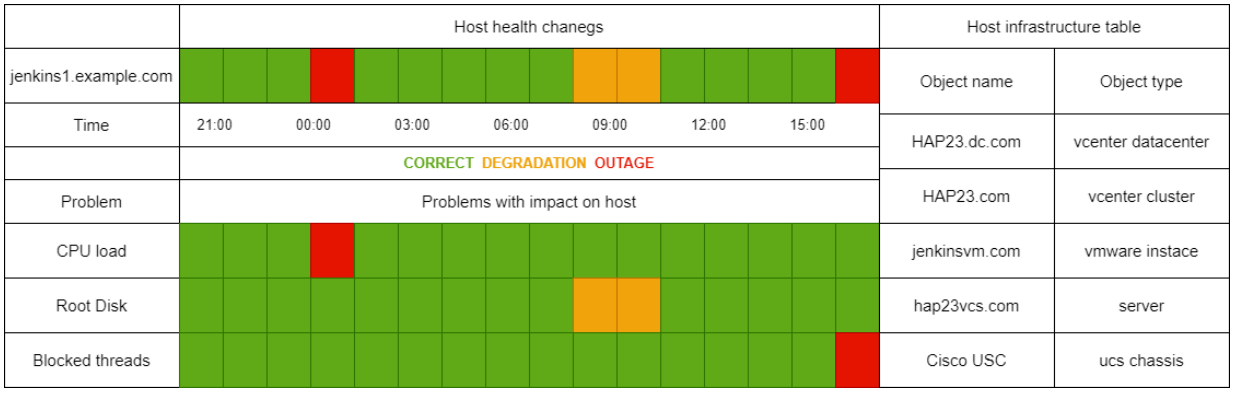

Fig. 6. Dashboard with service host detailed information

Fig. 6 shows a dashboard that provides detailed information about a service host with a list of its infrastructure elements and problems with them. The presented visualization allows you to find the sources of outages and identify vulnerable elements of the IT infrastructure that caused several failures.

Results

The described algorithm to improve the stability of the IT infrastructure was applied in one of the manufacturing companies with more than 10000 employees. Based on the information obtained with the help of the software we created, we managed to get recommendations on improving the stability of one of the services. In particular, one of the found problems was associated with simultaneous daily backups occurring on several hosts at once. The network file system could not cope with a large simultaneous load, which is why the service went into a degradation state.

Changes were applied at the end of January 2022: daily backups on each host started to be performed at different times, which reduced the load on the network file system. Table 2 shows the results of implementing the algorithm in manufacturing company. Prior to our changes, the service stability indicators

for January 2022 were as follows: availability 98.50%, reliability 92.58%. After the implementation of the developed software, the service stability indicators for February were as follows: availability 98.69%, reliability 94.43%.

Table 2

Results of implementing the algorithm in manufacturing IT-company

Metric	January 2022	February 2022	Relative change
Availability	98,50%	98,69%	0,19%
Reliability	92,58%	94,43%	2,00%
Absence	1,50%	1,31%	–12,67%
Fragility	7,42%	5,57%	–24,93%

Such an increase in metrics is a significant achievement, since even tenths and hundredths of a percent are important when agreeing on SLAs [21]. There are results that are even more impressive if we turn to the relative change of absence and fragility metrics. We managed to reduce the absence periods by 13% and fragility periods by 25% relative to the same values in the previous month. This shows a significant decrease of time during which the material and human resources of the company were idle.

Conclusion

In the course of the work, an algorithm was proposed to improve the stability of the IT infrastructure, based on the collection and analysis of data on its status. The proposed algorithm is based on two new metrics: availability and reliability, which were created based on a comprehensive study and comparative analysis of existing stability metrics. A distinctive feature of the new metrics is that they take into account the severity of failures in the system. The proposed software architecture of the system and its implementation in the software tool responsible for monitoring the stability of IT services made it possible to detect vulnerable elements of the IT infrastructure in an enterprise manufacturing company. After eliminating the vulnerabilities found, the relative decrease of instability periods was recorded using two metrics at once: the absence time of the service decreased by 13%, and the fragility time decreased by 25%. Such significant decrease of time during which the material and human resources of the company were idle shows the effectiveness of the developed algorithm.

REFERENCES

1. **Drobintsev P.D., Kotlyarova L.P., Voinov N.V., Tolstoles A.A., Maslakov A.P., Krustaleva I.N.** Automating preparation of small-scale production for reliable net-centric IoT workshop. CEUR Workshop Proceedings: APSSE 2019, 2019, pp. 75–85.

2. **Lazareva N.B.** Organizatsiia otkazoustoichivogo (HA) klastera [Organization of high-availability (HA) cluster]. E-Scio.ru, 2021, no. 2 (53), pp. 61–66.

3. **Unagayev S.** Organizatsiia IT-infrastruktury kompanii [Organization of IT-infrastructure in a company]. Sistemnyi administrator [System administrator], 2013, no. 4 (125), pp. 40–41.

4. **Autenrieth P., Lörcher C., Pfeiffer C., Winkens T., Martin L.** Current Significance of IT-Infrastructure Enabling Industry 4.0 in Large Companies. 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), 2018, pp. 1–8. DOI: 10.1109/ICE.2018.8436244

5. **Wang G., Zhang W., Huang C., Chen Z.** Service Availability Monitoring and Measurement Based on Customer Perception. 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), 2018, pp. 328–331. DOI: 10.1109/ICSESS.2018.8663889

6. **Ahmad A., Ngah L., Ibrahim I.** A Review of Service Quality Elements towards the Overlapping IT Framework Process on the IT Hardware Support Services (ITHS). International Journal of Advanced Trends in Computer Science and Engineering, 2020, Vol. 9, no. 1.4, pp. 423–432. DOI: 10.30534/ijatcse/2020/6091.42020

7. **Pan C., Xu H., Li W., Tu Z., Xu X., Wang Z.** Quality Monitoring and Measuring for Internet of Services. 2021 International Conference on Service Science (ICSS), 2021, pp. 107–114. DOI: 10.1109/ICSS53362.2021.00025

8. **Alimi O.A., Ouahada K., Abu-Mahfouz A.M.** A Review of Machine Learning Approaches to Power System Security and Stability. IEEE Access, 2020, vol. 8, pp. 113512–113531. DOI: 10.1109/ACCESS.2020.3003568

9. **He S., Lin Q., Lou J., Zhang H., Lyu M., Zhang D.** Identifying impactful service system problems via log analysis. ESEC/FSE 2018: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2018, pp. 60–70. DOI: 10.1145/3236024.3236083

10. **Bukhsh M., Abdullah S., Bajwa I.S.** A Decentralized Edge Computing Latency-Aware Task Management Method With High Availability for IoT Applications. IEEE Access, 2021, vol. 9, pp. 138994–139008. DOI: 10.1109/ACCESS.2021.3116717

11. **Li L., Dong J., Zuo D., Wu J.** SLA-Aware and Energy-Efficient VM Consolidation in Cloud Data Centers Using Robust Linear Regression Prediction Model. IEEE Access, 2019, vol. 7, pp. 9490–9500. DOI: 10.1109/ACCESS.2019.2891567

12. **Jiang C., Wang H., Yang Y., Fu Y.** Construction and Simulation of Failure Distribution Model for Cycloidal Gears Grinding Machine. IEEE Access, 2022, vol. 10, pp. 65126–65140. DOI: 10.1109/ACCESS.2022.3184318

13. **Li A., Han G., Ohtsuki T.** Multiple Radios for Fast Rendezvous in Heterogeneous Cognitive Radio Networks. IEEE Access, 2019, vol. 7, pp. 37342–37359. DOI: 10.1109/ACCESS.2019.2904942

14. **Al-Akwaa N.** The application of Lean Six Sigma principles in the technical support call center: First contact resolution. USA: California State University, Dominguez Hills, 2015. 144 p.

15. **Rukavitsyn A.N.** Data clustering in distributed monitoring systems. Informatsionno-upravliaiushchie sistemy [Information and Control Systems], 2019, no. 2, pp. 35–43. DOI: 10.31799/1684-8853-2019-2-35-43

16. **Huang Q., Xia X., Xing Z., Lo D., Wang X.** API Method Recommendation without Worrying about the Task-API Knowledge Gap. 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE), 2018, pp. 293–304. DOI: 10.1145/3238147.3238191

17. **Vázquez-Ingelmo A., Garcia-Peñalvo F.J., Therón R.** Information Dashboards and Tailoring Capabilities – A Systematic Literature Review. IEEE Access, 2019, vol. 7, pp. 109673–109688. DOI: 10.1109/ACCESS.2019.2933472

18. **Kovalev A.D., Voinov N.V., Nikiforov I.V.** Using the Doc2Vec Algorithm to Detect Semantically Similar Jira Issues in the Process of Resolving Customer Requests. Studies in Computational Intelligence, 2020, Vol. 868, pp. 96–101. DOI 10.1007/978-3-030-32258-8_11

19. **Voinov N., Rodriguez Garzon K., Nikiforov I., Drobintsev P.** Big data processing system for analysis of GitHub events. Proceedings of 2019 22nd International Conference on Soft Computing and Measurements (SCM 2019), 2019, pp. 187–190. DOI: 10.1109/SCM.2019.8903782

20. **Perrot A., Bourqui R., Hanusse N., Auber D.** HeatPipe: High Throughput, Low Latency Big Data Heatmap with Spark Streaming. 2017 21st International Conference Information Visualisation (IV), 2017, pp. 66–71. DOI: 10.1109/iV.2017.45

21. **Zharinova O.V.** Improving the efficiency of IT infrastructure management using ITIL. StudNet, 2021, vol. 4, no. 2, p. 43.

INFORMATION ABOUT AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

Varlamov Dmitrii A.
Варламов Дмитрий Андреевич
E-mail: varlamov_dmitry99@mail.ru

Nikiforov Igor V.
Никифоров Игорь Валерьевич
E-mail: igor.nikiforovv@gmail.com
ORCID: https://orcid.org/0000-0003-0198-1886

Ustinov Sergey M.
Устинов Сергей Михайлович
E-mail: usm50@yandex.ru
ORCID: https://orcid.org/0000-0003-4088-4798

Submitted: 03.06.2024; Approved: 24.07.2024; Accepted: 30.07.2024.
Поступила: 03.06.2024; Одобрена: 24.07.2024; Принята: 30.07.2024.