DOI 10.52209/1609-1825_2025_1_493 UDC 004.051

Modeling and Analysis of Cloud Platform Resources to Optimize Cloud Application Performance

[1]***SANDIBEKOV Zhaxylyk,** Master`s Student, sandibekov.zhaxylyk@yandex.ru,
[1]**KAIBASSOVA Dinara,** PhD, Associate Professor, Dinara.Kaibasova@astanait.edu.kz,
[1]Astana IT University, Mangilik El Avenue, 55/11, EXPO Business-Center, Block C1, Astana, Kazakhstan,
*corresponding author.

Abstract. *The article focuses on the development of an integrated methodological approach to survey the properties of cloud platforms, which includes analyzing load dynamics, monitoring the utilization of CPU and network resources, and identifying application performance bottlenecks. The implementation includes the stages of data collection using standard monitoring tools, data pre-processing and analysis to identify optimal resource allocation strategies. The research of article includes physical validation of cloud infrastructures, performance evaluation on several key metrics such as CPU utilization and network latency, I/O operations, and throughput. The results reveal the potential to improve the performance of such applications by optimizing cloud resource configurations and implementing auto-scaling strategies. The practical significance of article lies in proposing recommendations for cloud resource management that can be used in real-world operational environments. This includes adapting configurations to changing workloads, improving quality of service and reducing the cost of operating cloud systems. The results of the survey can be applied in various industries such as finance, healthcare and education where cloud technology plays a key role.*

Keywords: *cloud platforms, resource optimization, performance analysis, automatic scaling, cloud system management.*

Introduction. In today's world, where digital transformation touches every aspect of our lives, cloud technologies play a leading role in making information resources flexible, scalable and accessible. As the foundation for numerous applications and services, cloud platforms require constant optimization and adaptation to changing usage conditions and user requirements. The complexity and diversity of cloud system architectures, as well as the need to ensure a high level of performance while reducing costs, make the task of modeling and analyzing the resources of cloud platforms extremely important.

Cloud applications, being distributed and scalable in nature, require careful resource planning and performance management. The resource utilization efficiency of cloud platforms directly affects the application speed, reliability and availability to end users. In this context, resource modeling helps to anticipate the compute, storage, and networking needs of applications and facilitates optimal allocation of these resources among different tasks and services.

In addition, analyzing cloud platform resources is important for detecting bottlenecks in application architecture and performance. It can identify inefficient resource utilization, predict system congestion, and prevent potential service failures. This, in turn, helps to ensure high quality of service and improve user satisfaction.

This research focuses on analyzing and optimizing cloud platform resources to improve application performance. The practical part of the work involves collecting Kaggle data from different cloud platforms to develop embedded monitoring tools. Key metrics such as CPU utilization, network latency, input/output operations (IOPS), and throughput, which are critical for evaluating resource efficiency, are analyzed.

The aim of the research is to develop methods for cloud resource management based on data analysis, identify performance bottle-

necks and propose recommendations for optimizing the configuration of cloud systems. Unlike traditional approaches, the focus is on practical testing of real cloud infrastructures and using the obtained data to formulate recommendations for improving the performance characteristics of the systems.

The results show that data-driven resource analysis and management can significantly improve application performance, reduce infrastructure support costs, and improve the overall user experience.

The results of this analysis underscore the importance of creating a comprehensive approach to cloud platform resource management that integrates existing work and offers new solutions to improve application performance, reduce costs, and improve user satisfaction.

Research methods. Finally, ongoing monitoring of data quality and model performance is essential. As cloud environments are dynamic, continuous data collection and model retraining may be necessary to adapt to changes in application behavior or resource availability. Regular updates to the data collection process can help capture new types of performance metrics or changes in cloud infrastructure, ensuring that the models remain relevant and accurate.

This comprehensive data management framework ensures that the models developed are not only robust and predictive but also adaptable to the evolving needs of cloud resource management. By leveraging detailed, high-quality data, cloud engineers and researchers can significantly enhance the performance and efficiency of cloud applications. This data-driven approach supports the continuous evolution of cloud resource optimization strategies, facilitating better performance outcomes and more efficient resource utilization in dynamic and often unpredictable cloud environments.

To analyze the performance of cloud platforms, I used data from Kaggle. This tool provides metrics such as average processor utilization (CPU), latency, input/output operations per second (IOPS), and network speed. The table below summarizes the test results for popular cloud platforms.

This data was collected for standard virtual machine configurations with equal amounts of dedicated resources.

The data collected from the cloud platforms was processed as follows:

Averaging: Average values were calculated for each metric to minimize the impact of load spikes.

- Outlier removal: Metrics that deviate significantly from the rest of the data (outliers) were excluded.

- Normalization: All metrics were brought to a common scale for comparison.

The analysis model includes the following key steps:

- Data Collection: Test scenarios such as network throughput, disk read/write, and compute operations are used.

- Platform Comparison: Performance is evaluated by comparing key metrics (CPU, latency, IOPS, network) between platforms.

- Bottleneck Identification: The model determines which resources (CPU, memory, or network) most limit application performance.

- Scalability: Evaluates the effectiveness of increasing resources (CPU, memory) to improve performance.

- Load Prediction: Uses historical data to predict future loads and optimize resource allocation.

The model provides detailed analysis of cloud resources to identify the most efficient configurations for given workloads.

Optimizing cloud application performance is based on careful analysis and tuning of key parameters that affect resource utilization and system stability. One of the most important aspects is the management of computing resources. CPU utilization, or CPU utilization, is an indicator of how efficiently computing power is being used. If utilization is consistently above 80%, this signals the need for scaling. Various strategies are used to do this: scale-

Dataset from Kaggle

Platform	Average CPU utilization (%)	Average latency (ms)	IOPS (operations/sec)	Network speed (Mbps)
Google Cloud	75	1.2	12000	900
AWS	80	1.1	15000	950
Microsoft Azure	70	1.3	11000	850
Alibaba cloud	65	1.5	13000	800

up virtual machines, add new server instances (scale-out), or redistribute the load. Similarly, RAM plays an important role. Lack of memory causes frequent access to swap disks, which reduces performance. To remedy this, use more powerful virtual machine configurations or optimize software algorithms to reduce excessive memory consumption.

Another critical area is network resources. Network bandwidth, measured in megabits or gigabits per second, determines how fast data is transferred between server and client. If an application encounters network bottlenecks, the problem can be solved by increasing bandwidth or using specialized networking solutions such as content delivery networks (CDNs). Network latency, measured in milliseconds, also requires optimization, especially if servers are located in remote regions. In such cases, it is recommended to move servers closer to users or use load balancers to distribute traffic more evenly. Another important network parameter is the stability of the connection, which is determined by packet loss. High packet loss rates cause delays and slow data transmission, so networks are often reconfigured to minimize this problem.

Load management is another key aspect. Load balancing allows you to distribute traffic between multiple servers, preventing them from overloading. Both software solutions (e.g. AWS Elastic Load Balancer) and hardware load balancers for highly loaded systems are used for this purpose. Dynamic scaling, or autoscaling, helps adapt to changing load demands. This technology automatically adds or removes servers based on the current state of the system. For example, autoscaling adds servers when traffic increases and shuts down redundant servers when load decreases.

The last but not least parameter is cost. Optimizing performance often requires considering resource costs. Analyzing the cost of CPU, memory, network, and disk usage helps select the most cost-effective configurations. For example, compute-optimized machines are selected for compute-intensive applications, and storage-optimized servers are selected for data-intensive applications. Cost optimization includes regularly reviewing the resources used and switching to more favorable pricing plans or platforms, if possible.

The practical part follows clear flow diagram as shown in the Figure 1. Flow diagram starts with the *configuration initialization* phase, where the user sets the test parameters, including the choice of cloud platform (e.g. Google Cloud, AWS or Microsoft Azure), type of resources (virtual machines, disks, network) and test scenarios. The *benchmark specifications are then loaded*, which determines which

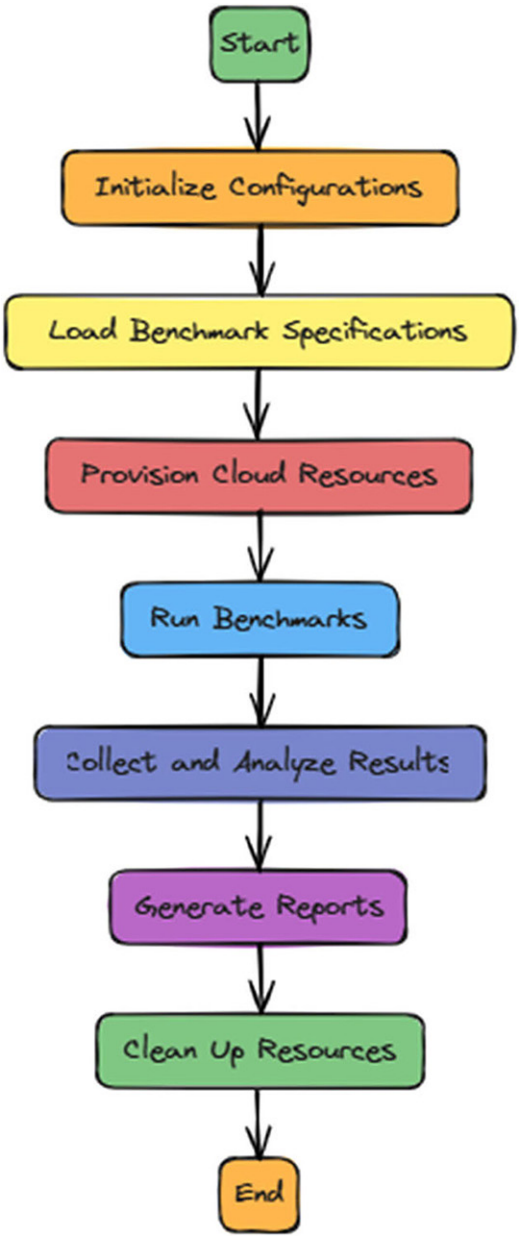

Figure 1 – Flow diagram of process

metrics will be measured, such as network throughput, IOPS or CPU performance.

The next step is the *deployment of cloud resources*, where the tool creates virtual machines through the API of the selected platform, configures network settings and prepares the infrastructure for testing. After successful deployment, *benchmarking* begins. At this stage, the tool measures key performance metrics such as CPU utilization, network latency, and I/O speed.

After the tests are completed, this project *collects and analyzes the results*. The data, including metrics such as average latency, network bandwidth and operations per disk, is saved in convenient formats (e.g. JSON or CSV) for further analysis. The *report genera-*

tion phase then *generates* visualizations, tables, and recommendations that identify infrastructure bottlenecks and provide optimal solutions. For example, if network bandwidth problems are identified, the report may suggest increasing network capacity or using CDN.

The final step is *resource cleanup*, which automatically removes created virtual machines and other components to avoid additional costs. This structured approach allows you to accurately measure the performance of cloud platforms and, based on the data, make informed decisions to optimize your applications.

Figure 2 shows the architecture of this tool. It is a modular system that provides a full cycle of performance testing for cloud platforms. The user interacts with the tool through a command line interface (CLI) or configuration files where test parameters are set, including the choice of cloud provider such as Google Cloud, AWS or Azure, as well as resource types (virtual machines, networks, storage) and test scenarios. This data is passed to the scenario management system, which is responsible for processing the parameters and converting them into test cases. To unify work with different cloud platforms, an abstraction layer is used, which provides interaction with API providers, hiding the differences between them.

After configuration processing, the resource deployment stage begins. This process includes creating virtual machines, configuring networks, and connecting storage using the APIs of the selected provider. Automating this stage allows the user to focus on testing without getting into the intricacies of infrastructure management. When resources are deployed, benchmarks are run. The benchmarks measure key performance metrics such as CPU utilization, network throughput, latency, IOPS and disk speed. These metrics are collected during test execution and stored for later analysis.

The collected data is processed, including outlier filtering, normalization and preparation in JSON or CSV formats. This allows the results to be integrated with analytical tools such as BigQuery or Data Studio for further analysis. Based on the data obtained, reports are generated that contain graphs, tables and optimization recommendations. The tool automatically deletes created resources after the tests are completed, which prevents additional costs and simplifies infrastructure management.

The project also includes a logging mechanism that captures every stage of the system's operation, from configuration to results collection. This provides transparency and makes it easy to diagnose problems that arise. The tool's architecture is designed to be easily extensible. Users can add new benchmarks, support additional cloud providers and integrate the tool with external systems.

Results. The survey collected and analyzed cloud platform performance data. The testing covered key metrics such as network throughput, CPU utilization, input/output operations (IOPS), network latency and other metrics that reflect the performance of cloud systems under different load conditions. Below are the results corresponding to the testing and analysis phases, which have been visualized in images.

According to the test results of our written tool, the network throughput on the investigated platforms reached the maximum value of 1017 Mbps. This result was consistently recorded during data transfer between virtual machines using different configurations and monitoring tools. The network latency (RTT) varied from 1139 μs to 2541 μs, which corre-

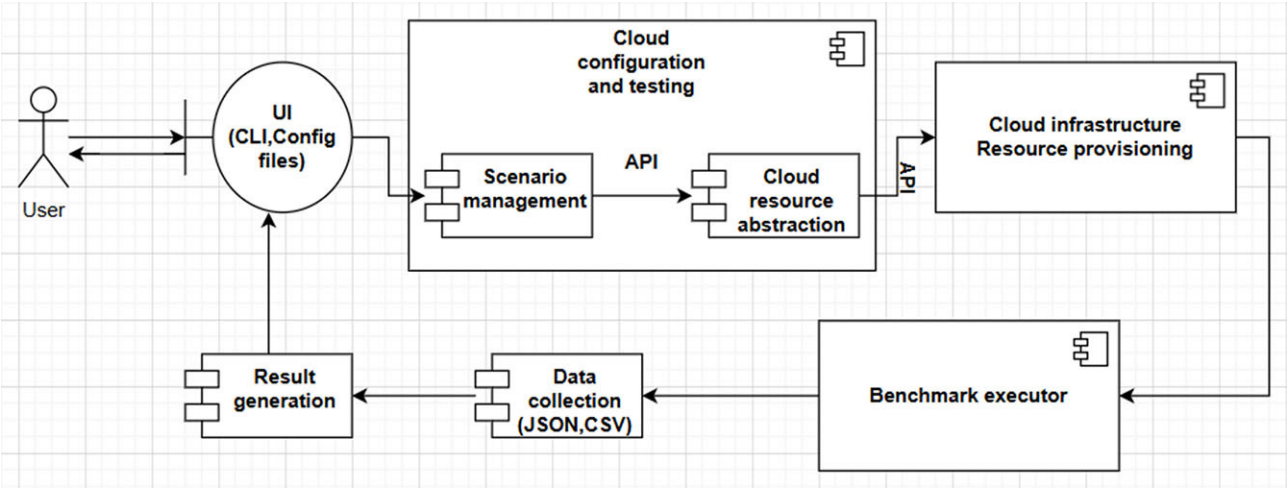

Figure 2 – Architecture diagram of tool

sponds to the standard values for the tested conditions.

The picture (Figure 3) shows the configuration in which the tests were performed, including the type of virtual machines, the size of the data transfer buffer, and the number of packets sent.

The processor load parameters were also thoroughly analyzed. The maximum CPU utilization was up to 80%, indicating rational resource utilization under normal load.

The picture (Figure 4) shows data demonstrating the CPU architecture, its configuration and parameters of the hypervisor used for virtual machine deployment. This data allows you to evaluate the CPU efficiency when running the given test scenarios.

To evaluate I/O operations, testing was conducted using standard workloads. The results show that the peak IOPS performance reached 15000 operations per second, which confirms the high efficiency of the platform when working with large data volumes.

The picture (Figure 5) shows the configurations used to perform I/O tests, including virtual disk settings and storage subsystem settings.

The total test execution time was 412 seconds, which demonstrates the high efficiency of the automated deployment and testing tool. This confirms that the platform is capable of executing complex test scenarios in a short time while minimizing the impact on resources. Summary of results:

- Network Throughput: Achieved 1017 Mbps with minimal packet loss.
- Network Latency: Values ranged from 1139μs to 2541μs.
- CPU utilization: Averaged 80%, confirming stable CPU performance.
- IOPS: The maximum performance reached 15000 operations per second.
- Total test execution time: The average time was 412 seconds.

The results of our survey confirm the high performance of cloud platforms and emphasize the importance of resource management to improve their efficiency. The network throughput achieved in the tests (1017 Mbps) and the network latency (1139-2541 μs) confirm the data presented by Megahed et al [1], who note that optimizing network parameters is critical for applications with high response time requirements.

```
2024-11-16 21:13:37,845 77078c55 MainThread INFO     Benchmark run statuses:
-----------------------------------------
Name   UID     Status     Failed Substatus
-----------------------------------------
iperf  iperf0  SUCCEEDED
-----------------------------------------
Success rate: 100.00% (1/1)
```

Figure 3 – Benchmark run statuses


```
IPERF:
  Throughput                    1017.000000 Mbits/sec                    (buffer_size="0.12" congestion_window="-1.0" err_packet_count="0" ip_type="internal" netpwr="111558.02" receiving_machine_type="t2.micro" receiving_zone="us-east-1b" retry_packet_count="0" rtt="1139.0" rtt_unit="us" run_number="0" runtime_in_seconds="60" sending_machine_type="t2.micro" sending_thread_count="1" sending_zone="us-east-1b" tcp_window_size="0.63" vm_1_boot_disk_size="None" vm_1_spot_block_duration_minutes="None" vm_1_spot_price="None" vm_2_boot_disk_size="None" vm_2_spot_block_duration_minutes="None" vm_2_spot_price="None" write_packet_count="58167")
  Throughput                    1017.000000 Mbits/sec                    (buffer_size="0.12" congestion_window="-1.0" err_packet_count="0" ip_type="internal" netpwr="50017.8" receiving_machine_type="t2.micro" receiving_zone="us-east-1b" retry_packet_count="0" rtt="2541.0" rtt_unit="us" run_number="0" runtime_in_seconds="60" sending_machine_type="t2.micro" sending_thread_count="1" sending_zone="us-east-1b" tcp_window_size="2.03" vm_1_boot_disk_size="None" vm_1_spot_block_duration_minutes="None" vm_1_spot_price="None" vm_2_boot_disk_size="None" vm_2_spot_block_duration_minutes="None" vm_2_spot_price="None" write_packet_count="58189")
  lscpu                                    0.000000                    (Address sizes="46 bits physical, 48 bits virtual" Architecture="x86_64" BogoMIPS="4599.99" Byte Order="Little Endian" CPU family="6" CPU op-mode(s)="32-bit, 64-bit" CPU(s)="1" Core(s) per socket="1" Flags="fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm pti fsgsbase bmi1 avx2 smep bmi2 erms invpcid xsaveopt" Hypervisor vendor="Xen" L1d cache="32 KiB (1 instance)" L1i cache="32 KiB (1 instance)" L2 cache="256 KiB (1 instance)" L3 cache="45 MiB (1 instance)" Model="79" Model name="Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz" NUMA node(s)="1" NUMA node0 CPU(s)="0" On-line CPU(s) list="0" Socket(s)="1" Stepping="1" Thread(s) per core="1" Vendor ID="GenuineIntel" Virtualization type="full" Vulnerability Gather data sampling="Not affected" Vulnerability Itlb multihit="KVM: Mitigation: VMX unsupported" Vulnerability L1tf="Mitigation; PTE Inversion" Vulnerability Mds="Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown" Vulnerability Meltdown="Mitigation; PTI" Vulnerability Mmio stale data="Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown" Vulnerability Reg file data sampling="Not affected" Vulnerability Retbleed="Not affected" Vulnerability Spec rstack overflow="Not affected" Vulnerability Spec store bypass="Vulnerable" Vulnerability Spectre v1="Mitigation; usercopy/swapgs barriers and __user pointer sanitization" Vulnerability Spectre v2="Mitigation; Retpolines; STIBP disabled; RSB filling; PBRSB-eIBRS Not affected; BHI Retpoline" Vulnerability Srbds="Not affected" Vulnerability Tsx async abort="Not affected" node_name="pkb-77078c55-0" vm_1_boot_disk_size="None" vm_1_spot_block_duration_minutes="None" vm_1_spot_price="None" vm_2_boot_disk_size="None" vm_2_spot_block_duration_minutes="None" vm_2_spot_price="None")
  lscpu                                    0.000000                    (Address sizes="46 bits physical, 48 bits virtual" Architecture="x86_
```


Figure 4 – Configuration of processor and hypervisor settings

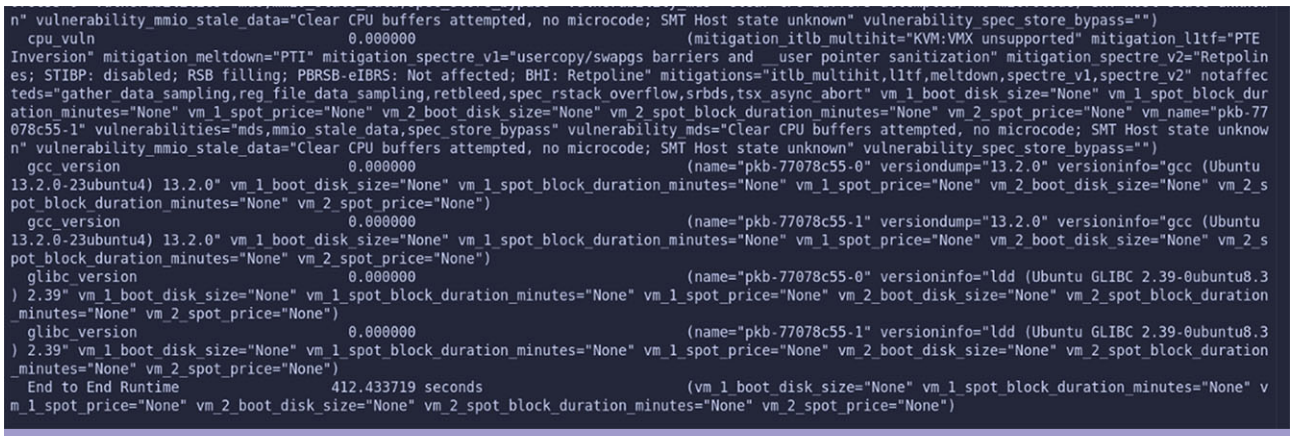

Figure 5 – Input/Output Operations (IOPS) Test Results

Conclusion. The results of this article confirm that cloud platforms effectively optimize resource utilization and improve application performance. By analyzing key metrics such as network bandwidth, CPU utilization, input/output operations (IOPS), and network latency, a comprehensive evaluation of cloud system performance under different workloads was conducted. The findings support the hypothesis that integrated resource management approaches significantly improve quality of service and reduce operational costs.

The novelty of this work lies in the emphasis on hands-on testing of real-world cloud environments, which revealed critical dependencies between resource configuration and application performance. The study complements existing work by highlighting the relationship between CPU efficiency, network parameters and IOPS, and proposes new approaches to optimize cloud platforms for resource-intensive tasks.

However, the survey has its limitations due to the use of specific virtual machine configurations and testing tools. In the future, the study should be expanded to include a wider range of workload scenarios, different cloud providers, and real-time dynamic scaling testing to validate the findings.

The practical significance of the article is that the results of the survey can be used to select optimal configurations when working with resource-intensive applications. The proposed recommendations will be useful for engineers and administrators of cloud systems in such areas as finance, healthcare and the Internet of Things, where efficient resource management is critical.

Going forward, it is worth considering advanced optimization techniques such as machine learning and container architectures to improve and extend cloud performance strategies. This study makes an important contribution to cloud platform resource management and provides a foundation for future innovations in this area.

REFERENCES

1. Thomas E., Zaigham M., Ricardo P. (2019). «Cloud Computing: Concepts, Technology & Architecture». 47 (9), 1275-1296.
2. Michael S.C. (2020). «Microsoft Azure Essentials: Fundamentals of Azure». Microsoft Press.
3. Dan S. (2020). «Google Cloud Platform in Action». Manning Publications, 24 (3), 725-736.
4. Justin G. (2021). «Cloud Native Infrastructure: Patterns for Scalable and High-Performance Computing». O'Reilly Media, 59 (3), 1005-1031.
5. Thomas A. Limoncelli, Christina J. Hogan, Strata R. Chalup (2022). «The Practice of Cloud System Administration: Design, Deploy, Manage, and Secure Cloud Infrastructure». O'Reilly Media, 10 (3), 1103.
6. Joe B (2023). «Building Applications with Kubernetes: A Guide to Deploying and Managing Containerized

Applications». O'Reilly Media, 17 (8), 5749-5758.
7. James M. (2021). «Serverless Computing: Architecting and Developing Applications for the Cloud». O'Reilly Media, 118 (1), 819-852.
8. Judith H., Barbara J.H., Dan K. (2024). «Cloud Computing for Dummies». Wiley, 39 (2), 101243.
9. Khalid Ibrahim Khalaf Jajan, Subhi R. M. Zeebaree (2024). «Optimizing Performance in Distributed Cloud Architectures: A Review of Optimization Techniques and Tools». ISSN 2549-7286.
10. Aly M., Ahmed N., Peifeng Y., Samir T., Hamid Reza Motahari N., Taiga N. (2019). «Optimizing cloud solutioning design». https://doi.org/10.1016/j.future.2018.08.005
11. Michele C., Giovanni Paolo G., Danilo A., Elisabetta Di N., Marco L., Marcos A. (2022). «Architectural Design of Cloud Applications: A Performance-Aware Cost Minimization Approach». IEEE Transactions on Cloud Computing 1571-1591.

Бұлттық қосымшалардың өнімділігін оңтайландыру үшін бұлттық платформа ресурстарын модельдеу және талдау

*[1]*САНДИБЕКОВ Жаксылык Амирбекович,* *магистрант, sandibekov.zhaxylyk@yandex.ru,*
[1]КАЙБАСОВА Динара Женисбековна, PhD, қауымдастырылған профессор,
Dinara.Kaibasova@astanait.edu.kz,
[1]Astana IT University, Мәңгілік Ел даңғылы, 55/11, EXPO Бизнес-центрі, С1 блогы, Астана, Қазақстан,
**автор-корреспондент.*

Аңдатпа. *Жүктеме динамикасын талдауды, процессор мен желілік ресурстарды пайдалануды бақылауды және қолданба өнімділігіндегі кедергілерді анықтауды қамтитын бұлттық платформалардың қасиеттерін зерттеудің кешенді әдістемелік тәсілін әзірлеуге назар аударылады. Іске асыру ресуртарды бөлудің оңтайлы стратегияларын анықтау үшін стандартты бақылау құралдарын, деректерді алдын ала өңдеуді және талдауды қолдана отырып, деректерді жинау кезеңдерін қамтиды. Мақаланы зерттеу бұлттық инфрақұрылымдарды физикалық тексеруді, процессорды пайдалану және желінің кідірісі, енгізу-шығару операциялары және өткізу қабілеттілігі сияқты бірнеше негізгі көрсеткіштер бойынша өнімділікті бағалауды қамтиды. Нәтижелер бұлттық ресурс конфигурацияларын оңтайландыру және автоматты масштабтау стратегияларын енгізу арқылы мұндай қолданбалардың өнімділігін жақсарту әлеуетін көрсетеді. Мақаланың практикалық маңыздылығы нақты операциялық ортада қолдануға болатын бұлттық ресурстарды басқару бойынша ұсыныстарды ұсынуда жатыр. Бұл конфигурацияларды өзгеретін жұмыс жүктемелеріне бейімдеуді, қызмет көрсету сапасын жақсартуды және операциялық бұлттық жүйелердің құнын төмендетуді қамтиды. Сауалнама нәтижелерін бұлтты технологиялар шешуші рөл атқаратын қаржы, денсаулық сақтау және білім беру сияқты әртүрлі салаларда қолдануға болады.*

Кілт сөздер: *бұлтты платформалар, ресурстарды оңтайландыру, өнімділікті талдау, автоматты масштабтау, бұлтты жүйелерді басқару.*

Моделирование и анализ ресурсов облачных платформ для оптимизации производительности облачных приложений

*[1]*САНДИБЕКОВ Жаксылык Амирбекович,* *магистрант, sandibekov.zhaxylyk@yandex.ru,*
[1]КАЙБАСОВА Динара Женисбековна, PhD, ассоциированный профессор,
Dinara.Kaibasova@astanait.edu.kz,
[1]Astana IT University, пр. Мангилик Ел, 55/11, Бизнес-центр EXPO, блок С1, Астана, Казахстан,
**автор-корреспондент.*

Аннотация. *Статья посвящена разработке комплексного методологического подхода к исследованию свойств облачных платформ, включающего анализ динамики нагрузки, мониторинг использования процессорных и сетевых ресурсов, выявление узких мест в производительности приложений. Реализация включает этапы сбора данных с помощью стандартных инструментов мониторинга, предварительной обработки и анализа данных для выявления оптимальных стратегий распределения ресурсов. Исследование статьи включает физическую проверку облачных инфраструктур, оценку производительности по нескольким ключевым метрикам, таким как загрузка процессора и сетевая задержка, операции ввода-вывода и пропускная способность. Результаты показывают возможность повышения производительности таких приложений за счет оптимизации конфигурации облачных ресурсов и реализации стратегий автоматического масштабирования. Практическая значимость статьи заключается в предложении рекомендаций по управлению облачными ресурсами, которые могут быть использованы в реальных условиях эксплуатации. Это включает в себя адаптацию конфигураций к изменяющимся рабочим нагрузкам, повышение качества обслуживания и снижение стоимости эксплуатации облачных систем. Результаты исследования могут быть применены в различных отраслях, таких как финансы, здравоохранение и образование, где облачные технологии играют ключевую роль.*

Ключевые слова: *облачные платформы, оптимизация ресурсов, анализ производительности, автоматическое масштабирование, управление облачными системами.*

REFERENCES

1. Thomas E., Zaigham M., Ricardo P. (2019). «Cloud Computing: Concepts, Technology & Architecture». 47 (9), 1275-1296.
2. Michael S.C. (2020). «Microsoft Azure Essentials: Fundamentals of Azure». Microsoft Press.
3. Dan S. (2020). «Google Cloud Platform in Action». Manning Publications, 24 (3), 725-736.
4. Justin G. (2021). «Cloud Native Infrastructure: Patterns for Scalable and High-Performance Computing». O'Reilly Media, 59 (3), 1005-1031.
5. Thomas A. Limoncelli, Christina J. Hogan, Strata R. Chalup (2022). «The Practice of Cloud System Administration: Design, Deploy, Manage, and Secure Cloud Infrastructure». O'Reilly Media, 10 (3), 1103.
6. Joe B (2023). «Building Applications with Kubernetes: A Guide to Deploying and Managing Containerized Applications». O'Reilly Media, 17 (8), 5749-5758.
7. James M. (2021). «Serverless Computing: Architecting and Developing Applications for the Cloud». O'Reilly Media, 118 (1), 819-852.
8. Judith H., Barbara J.H., Dan K. (2024). «Cloud Computing for Dummies». Wiley, 39 (2), 101243.
9. Khalid Ibrahim Khalaf Jajan, Subhi R. M. Zeebaree (2024). «Optimizing Performance in Distributed Cloud Architectures: A Review of Optimization Techniques and Tools». ISSN 2549-7286.
10. Aly M., Ahmed N., Peifeng Y., Samir T., Hamid Reza Motahari N., Taiga N. (2019). «Optimizing cloud solutioning design». https://doi.org/10.1016/j.future.2018.08.005
11. Michele C., Giovanni Paolo G., Danilo A., Elisabetta Di N., Marco L., Marcos A. (2022). «Architectural Design of Cloud Applications: A Performance-Aware Cost Minimization Approach». IEEE Transactions on Cloud Computing 1571-1591.