INFORMATION ON UTILIZATION OF DATA CENTER RESOURCES WITH MESSAGE BROKER IMPLEMENTATION

N.Yu. Samokhin[a], S.E. Khoruzhnikov[a], V.M. Trubnikova[a], R.R. Akhmedzyanova[b], A.B. Bulykina[b]

[a] ITMO University, Saint Petersburg, 197101, Russian Federation
[b] "Hostland", Saint Petersburg, 197342, Russian Federation
Corresponding author: samon@vuztc.ru

Article info
Received 28.04.18, accepted 25.05.18
doi: 10.17586/2226-1494-2018-18-5-858-862
Article in English

For citation: Samokhin N.Yu., Khoruzhnikov S.E., Trubnikova V.M., Akhmedzyanova R.R., Bulykina A.B. Information on utilization of data center resources with message broker implementation. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2018, vol. 18, no. 5, pp. 858–862 (in Russian). doi: 10.17586/2226-1494-2018-18-5-858-862

Abstract

A cloud of geographically distributed data centers developed by ITMO University (ifmo.ru) is presented. To increase reliability, as well as to collect statistical data, a special monitoring system is created that monitors the state of systems and the utilization of hardware resources. In addition to building a map of statistical data, information on resource utilization is suggested for the provider to perform a billing. An approach to monitoring system with asynchronous communication is proposed. An option with message broker implementation is shown. Out of two basic billing models, billing by quotas and billing by actual usage, we propose a model for organizing a dialogue between the administrator and the monitoring system in order to obtain data on resource utilization by actual usage. The use of a database for storing requests and responses is proposed, as well as the asynchronous dialogue using queues technology and message broker. Approbation was carried out on testing equipment with an actual infrastructure simulation. Unique asynchronous multi-agent system was obtained for statistics gathering on data center resources utilization. RabbitMQ message broker, PostgreSQL database and Zabbix monitoring system implementations are shown. The dialogue system was optimized with the help of Python code. The designed model can be recommended for implementation aimed at utilization of data obtained in different data centers, including distributed ones. Suggested model can be used in high-loaded systems providing virtual resources as a service.

Keywords
data center, virtualization, monitoring, billing, message broker, Zabbix, RabbitMQ

Acknowledgements
The research was carried out with the financial support of the Ministry of Education and Science of the Russian Federation (Contract No. 03.G25.31.0229).

УДК 004.75

ИНФОРМАЦИЯ ПО УТИЛИЗАЦИИ РЕСУРСОВ ЦЕНТРА ОБРАБОТКИ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ СИСТЕМЫ ОЧЕРЕДЕЙ

Н.Ю. Самохин[a], С.Э. Хоружников[a], В.М. Трубникова[a], Р.Р. Ахмедзянова[b], А.Б. Булыкина[b]

[a] Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация
[b] ООО «Хостланд», Санкт-Петербург, 197342, Российская Федерация
Адрес для переписки: samon@vuztc.ru

Информация о статье
Поступила в редакцию 01.06.18, принята к печати 31.07.18
doi: 10.17586/2226-1494-2018-18-5-858-862
Язык статьи – английский

Ссылка для цитирования: Самохин Н.Ю., Хоружников С.Э., Трубникова В.М., Ахмедзянова Р.Р., Булыкина А.Б. Информация по утилизации ресурсов центра обработки данных с использованием системы очередей // Научно-технический вестник информационных технологий, механики и оптики. 2018. Т. 18. № 5. С. 858–862. doi: 10.17586/2226-1494-2018-18-5-858-862

Аннотация

Представлена разрабатываемая Университетом ИТМО (ifmo.ru) облачная система для географически распределенных центров обработки данных. Для повышения отказоустойчивости, а также для сбора статистических данных создана специальная система мониторинга, которая отслеживает состояние систем и утилизацию аппаратных ресурсов. Помимо построения карты статистических данных, информация по утилизации ресурсов предлагается провайдеру для осуществления биллинга. Предложен подход с асинхронным обращением к системе мониторинга. Представлен вариант с использованием системы сообщений. Из двух основных моделей биллинга – биллинг по квотам и биллинг по фактическому использованию – предложена модель реализации диалога между администратором и системой мониторинга с целью получения данных об утилизации ресурсов по фактическому использованию. Предложено использование базы данных для хранения запросов и ответов, а также обеспечение асинхронности диалога с помощью технологии очередей и брокера сообщений. Апробация разработанной модели произведена на тестовом оборудовании с эмуляцией реальной инфраструктуры. Получена уникальная асинхронная мультиагентная система сбора статистических данных по использованию ресурсов центра обработки данных. Показано применение системы очередей RabbitMQ, базы данных PostgreSQL и системы мониторинга Zabbix. Система диалога оптимизирована с помощью языка программирования Python. Разработанная модель может быть рекомендована для использования при сборе информации по утилизации ресурсов в различных центрах обработки данных, в том числе распределенных. Предложенная модель может применяться в высоконагруженных системах, предоставляющих виртуальные ресурсы как сервис.

Ключевые слова
центр обработки данных, виртуализация, мониторинг, биллинг, очереди сообщений, Zabbix, RabbitMQ

Благодарности
Исследования выполнены при финансовой поддержке Министерства образования и науки Российской Федерации (Договор № 03.G25.31.0229).

Introduction

Data center is a set of server and network equipment and is designed to process, store and distribute information. Today, data centers usually operate in the interests of corporate clients; they are focused mostly on the provision of services for business issues. However, the use of data centers at certain points can be regarded as a risk of single points of failure. Therefore, the entire infrastructure must be connected to a monitoring system (MS). The use of MS provides a number of possibilities, such as:

- monitoring of the system state;
- monitoring of resource utilization;
- billing.

A billing gives the possibility to estimate the provided and utilized resources in terms of money. The main goal pursued by this work is to develop and propose a method of providing data for billing by querying the monitoring system in asynchronous mode. This method is not only progressive, but recommended greatly to be used in terms of information security [1–6]. Moreover, the method has its own advantages. First, it implements multi-agent system (also known as “self-organized system”). It denotes complete independence of all system components. Besides, several agent instances can be launched to provide high availability and reliability of the system. If there is an operation interruption, a user will not even notice it as multi-agent system is handling it.

Apart from that, relational database is being used. Such databases are often considered to be a bottleneck, so several actions have been taken to reduce its influence. Especially, the cluster system approach was implemented. It provided not only some high availability of the database, but load balancing as well. It is stressed that database primal function is to store requests and answer with the unique ID (UUID). Apart from that, it stores logging information.

This approach is considered to be unique. All mentioned components and tools are quite commonly used, but the joint set of them is thought to be progressive. There is hardly any mention of systems with message broker and relational database implementation at the same time where they are maintaining each other. It provides high reliability, load balancing and proper logging at every system component. Also, the usage of cluster systems of database and message broker contributes to the progress of the task. All facts mentioned above denote scientific novelty of this study. However, it seems to be difficult to compare suggested approach to the others because obtaining of statistical data has a close connection to monitoring system, and the latter often appears to be a closed system in developer infrastructure. Therefore, the approach being developed requires more of the own experience rather than the common one. That is what makes this model even more unique.

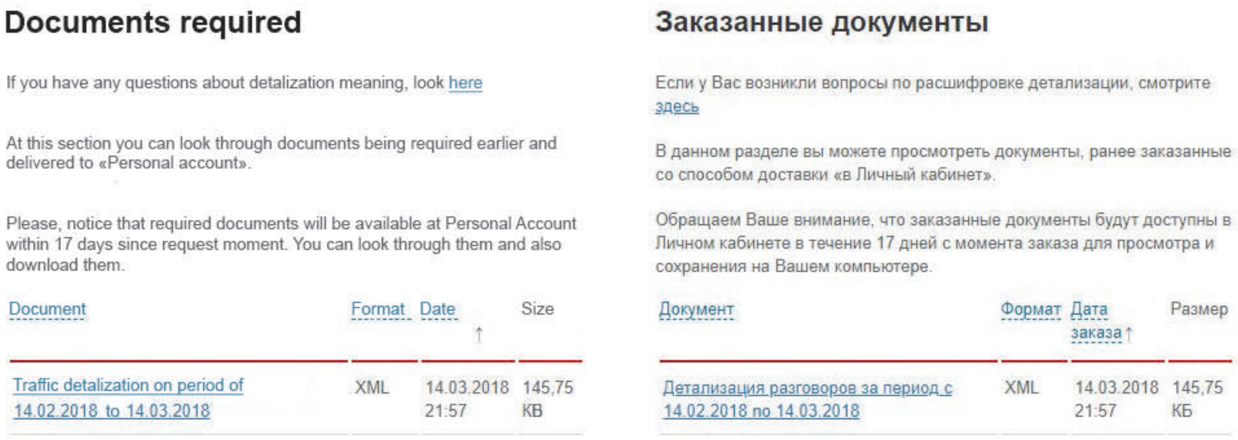
Description

An asynchronous mode assumes that the sender of the request is not required to support the session and wait for a response from the MS after the request has been made. The request for information on the data center (DC) resources utilization can be processed for a long time, depending, for example, on the date range of interest to the user [7–10]. One of the most known examples of this successfully implemented model is “Detailing” in the personal account settings of the Russian mobile provider MTS.

MTS brief model description includes the following steps:

- The user requires his (mobile) traffic detailing in his personal account environment;
- The system answers with “Request is processing, check the results in your account later”;
- In several minutes the formed detailing report can be obtained in personal account in any suitable format (or by e-mail).

An example of such model is presented in Figure 1.



Documents required

If you have any questions about detalization meaning, look here

At this section you can look through documents being required earlier and delivered to «Personal account».

Please, notice that required documents will be available at Personal Account within 17 days since request moment. You can look through them and also download them.

Document	Format	Date ↑	Size
Traffic detalization on period of 14.02.2018 to 14.03.2018	XML	14.03.2018 21:57	145,75 KB

a

Заказанные документы

Если у Вас возникли вопросы по расшифровке детализации, смотрите здесь

В данном разделе вы можете просмотреть документы, ранее заказанные со способом доставки «в Личный кабинет».

Обращаем Ваше внимание, что заказанные документы будут доступны в Личном кабинете в течение 17 дней с момента заказа для просмотра и сохранения на Вашем компьютере.

Документ	Формат	Дата заказа ↑	Размер
Детализация разговоров за период с 14.02.2018 по 14.03.2018	XML	14.03.2018 21:57	145,75 КБ

b

Figure 1. MTS “Detalization” in personal account settings in English (a) and in Russian (b)

There is the same approach we implemented here. The user asks for resources utilization data, and monitoring system gives him not the answer, but the link to an answer in special database (DB) in order to make the dialogue asynchronous.

To implement this model, we proposed to develop a special agent supposed to be in a waiting state, which will process the request from the user, send it to the monitoring system, and process the incoming response. Here, the agent is a program which is going to work like a Linux daemon and it is going to be an intermediator. The user can receive the response to the request in any form convenient for him. It is proposed to use a message broker system for communication between the user and the MS. Figure 2 illustrates the basic model.

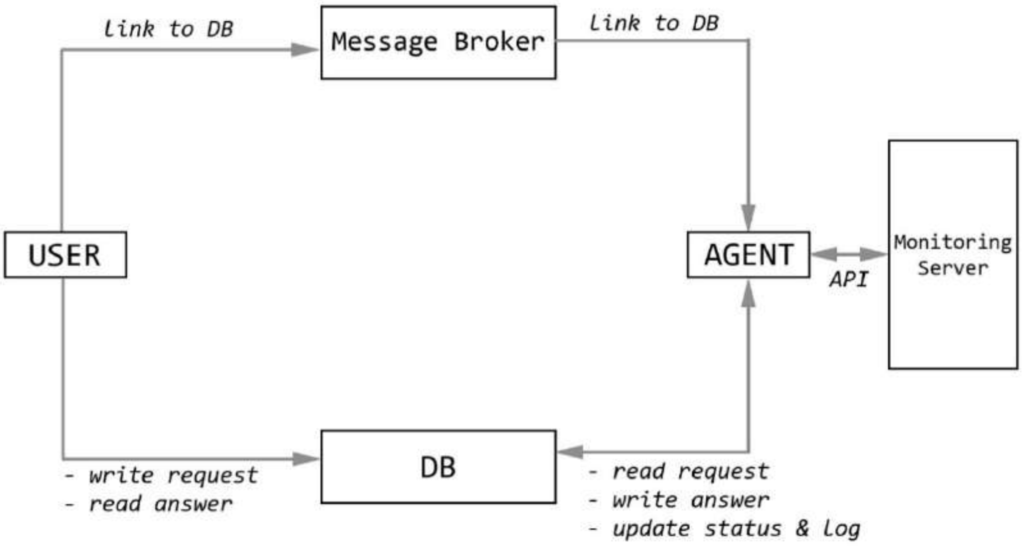


Figure 2. Basic model of user-MS dialogue

Development

An appropriate software should be chosen to reach our goals.

First of all, there is a monitoring system. Zabbix is known to be very flexible, scalable and convenient (API) so we decided to use it [11, 12]. Its components are:

- Zabbix Server 3.4: the core of MS;
- Zabbix Agent 3.4: a program (daemon) which collects monitoring data and sends it to Zabbix Server for processing and storing;
- Zabbix API for communication between Zabbix Server and special Linux agent mentioned in “Description” section.

It is also proposed to use a database for storing requests and responses, as well as for the logging [13]. Any relational database could be useful, so we have chosen PostgreSQL 9.6. Figure 3 shows a simple structure of the designed database. All entries use UUID as a primary key [14].

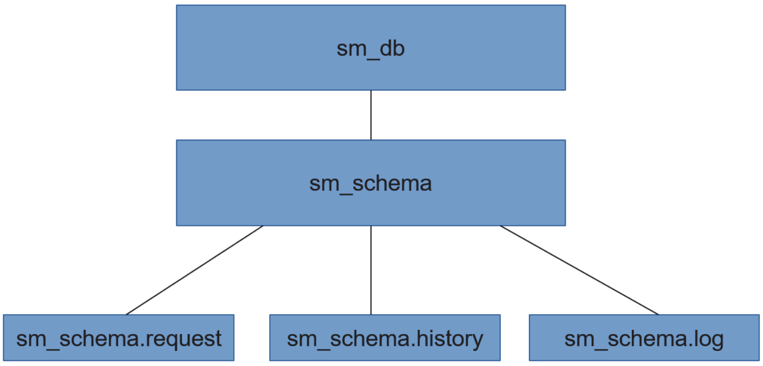


Figure 3. Basic structure of DB for user-MS dialogue

DB components are:
- “sm_db” as a DB name;
- “sm_schema” as a DB schema name;
- “request” as a table for storing user’s requests;
- “history” as a table for storing information about the user and the answer for his request;
- “log” as a table for storing logs.

Finally, the most important part is a mediator, i.e. the agent. It is a complex program which is supposed to run on Linux server as a daemon. It was decided to design it with the help of Python language, version 2.7.5 [15, 16].

RabbitMQ 3.6 was used as a messaging system [17].

Step-by-step description of operation is as follows:
1. A user is requesting his resources utilization data, for example, during this month. The answer is delivered immediately, containing “Your request is processing now. Check the answer later in sm_db.sm_schema.history”;
2. His request is sent to DB;
3. The Python Agent checks for new entries in DB, obtains them and starts processing;
4. The request is processed by the Agent, translated into appropriate syntax and sent to Zabbix Server by Zabbix API;
5. Zabbix Server answers with raw data by Zabbix API;
6. The Agent is processing raw data, translating it in syntax suitable for the user and sending it to DB;
7. The Agent goes back to standby (waiting) state.

Conclusion

This paper is a brief overview considering the implementation of asynchronous communication between the user and the monitoring system for various needs, the billing in particular. The proposed option eliminates the expectation of a response from the user by using the technology of message queues and an "interlayer" in the form of an agent. The agent works with the user, the monitoring system and the database, in which all requests and answers are recorded.

As this paper is being prepared, the model is in the alpha phase, but now the development is in the beta testing phase of the "The development of new technological components of geographically distributed DCs control systems, including resources virtualization (memory, communication lines, processing power, engineering infrastructure) with quantum technologies for communication lines protection" project of ITMO University. For today it is much more complex and stable. It has also passed all experiments and professional tests successfully in ITMO University environment. Unfortunately, there was no stressing testing of the approach, but the emphasis is made that it was designed to operate in high-load wide scalable environment with several datacenters being used.

Литература

1. Shevel A.Y., Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Kairkanov A. Globally distributed software defined storage (proposal) // Journal of Physics: Conference Series. 2017. V. 898. N 6. Art. 062014. doi: 10.1088/1742-6596/898/6/062014
2. Fedchenkov P.V., Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Shevel A.E., Kairkanov A.B., Lazo O.I., Oreshkin A.A. Service reliability in the cloud of data centers under Openstack // CEUR Workshop Proceedings. 2017. V. 2023. P. 282–287.
3. Shevel A.Y., Khoruzhnikov S.E., Grudinin V.A., Sadov O.L.,

References

1. Shevel A.Y., Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Kairkanov A. Globally distributed software defined storage (proposal). *Journal of Physics: Conference Series*, 2017, vol. 898, no. 6, art. 062014. doi: 10.1088/1742-6596/898/6/062014
2. Fedchenkov P.V., Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Shevel A.E., Kairkanov A.B., Lazo O.I., Oreshkin A.A. Service reliability in the cloud of data centers under Openstack. *CEUR Workshop Proceedings*, 2017, vol. 2023, pp. 282–287.

Kairkanov A.B. Geographically distributed software defined storage (the proposal) // CEUR Workshop Proceedings. 2016. V. 1787. P. 62–67.

4. Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Shevel A.E., Kairkanov A.B. Transfer of large volume data over Internet with parallel data links and SDN // Lecture Notes in Computer Science. 2015. V. 9142. P. 463–471. doi: 10.1007/978-3-319-20469-7_49
5. Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Shevel A.E., Titov V.B., Kairkanov A.B. Initial-stage examination of a testbed for the big data transfer over parallel links. The SDN approach // Astrophysical Bulletin. 2015. V. 70. N 2. P. 238–242. doi: 10.1134/S199034131502011X
6. Chen Y., Paxson V., Katz R. What’s New about Cloud Computing Security? Technical Report UCB/EECS-2010-5. Berkeley, 2009.
7. Greenberg A., Hamilton J.R., Jain N., Kandula S., Kim C., Lahiri P., Maltz D.A., Patel P., Sengupta S.VL2: scalable and flexible data center network // Computer Communication Review. 2009. V. 39. N 4. P. 51–62. doi: 10.1145/1594977.1592576
8. Mohd Saleem. Cloud computing virtualization // International Journal of Computer Applications Technology and Research. 2017. V. 6. N 7. P. 290–292.
9. Armbrust M., Fox A., Griffith R., Joseph A.D., Katz R., Konwinski A., Lee G., Patterson D., Rabkin A., Stoica I., Zaharia M. A view of cloud computing // Communications of the ACM. 2010. V. 53. N 4. P. 50–58. doi: 10.1145/1721654.1721672
10. Ali-Eldin A.H. Capacity Scaling for Elastic Compute Clouds. Dissertation. Umea University, Sweden, 2013. 90 p.
11. Liu S.Y. Research and implementation of automated inspection system based on ZABBIX // Electric Power Information and Communication Technology. 2014. V. 12. N 12. P. 111–115.
12. Dalle Vacche A. Mastering Zabbix. 2nd ed. Packt Publ., 2015. 412 p.
13. Elmasri R., Navathe S.B. Fundamentals of Database Systems. 6th ed. Addison Wesley, 2010. 1200 p.
14. Momjian B. PostgreSQL: Introduction and Concepts. Addison-Wesley, 2002. 490 p.
15. Dierbach C. Introduction to Computer Science Using Python: A Computational Problem-Solving Focus. Wiley, 2012. 612 p.
16. Millman K.J., Aivazis M. Python for scientists and engineers // Computing in Science and Engineering. 2011. V. 13. N 2. P. 9–13. doi: 10.1109/MCSE.2011.36
17. Rostanski M., Grochla K., Seman A. Evaluation of highly available and fault-tolerant middleware clustered architectures using RabbitMQ // Proc. 2014 Federated Conference on Computer Science and Information Systems. Warsaw, Poland, 2014. V. 2. P. 879–884. doi: 10.15439/2014F48

3. Shevel A.Y., Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Kairkanov A.B. Geographically distributed software defined storage (the proposal). *CEUR Workshop Proceedings*, 2016, vol. 1787, pp. 62–67.
4. Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Shevel A.E., Kairkanov A.B. Transfer of large volume data over Internet with parallel data links and SDN. *Lecture Notes in Computer Science*, 2015, vol. 9142, pp. 463–471. doi: 10.1007/978-3-319-20469-7_49
5. Khoruzhnikov S.E., Grudinin V.A., Sadov O.L., Shevel A.E., Titov V.B., Kairkanov A.B. Initial-stage examination of a testbed for the big data transfer over parallel links. The SDN approach. *Astrophysical Bulletin*, 2015, vol. 70, no. 2, pp. 238–242. doi: 10.1134/S199034131502011X
6. Chen Y., Paxson V., Katz R. What’s New about Cloud Computing Security? *Technical Report UCB/EECS-2010-5*. Berkeley, 2009.
7. Greenberg A., Hamilton J.R., Jain N., Kandula S., Kim C., Lahiri P., Maltz D.A., Patel P., Sengupta S.VL2: scalable and flexible data center network. *Computer Communication Review*, 2009, vol. 39, no. 4, pp. 51–62. doi: 10.1145/1594977.1592576
8. Mohd Saleem. Cloud computing virtualization. *International Journal of Computer Applications Technology and Research*, 2017, vol. 6, no. 7, pp. 290–292.
9. Armbrust M., Fox A., Griffith R., Joseph A.D., Katz R., Konwinski A., Lee G., Patterson D., Rabkin A., Stoica I., Zaharia M. A view of cloud computing. *Communications of the ACM*, 2010, vol. 53, no. 4, pp. 50–58. doi: 10.1145/1721654.1721672
10. Ali-Eldin A.H. *Capacity Scaling for Elastic Compute Clouds*. Dissertation. Umea University, Sweden, 2013, 90 p.
11. Liu S.Y. Research and implementation of automated inspection system based on ZABBIX. *Electric Power Information and Communication Technology*, 2014, vol. 12, no. 12, pp. 111–115.
12. Dalle Vacche A. *Mastering Zabbix*. 2nd ed. Packt Publ., 2015, 412 p.
13. Elmasri R., Navathe S.B. *Fundamentals of Database Systems*. 6th ed. Addison Wesley, 2010, 1200 p.
14. Momjian B. *PostgreSQL: Introduction and Concepts*. Addison-Wesley, 2002, 490 p.
15. Dierbach C. *Introduction to Computer Science Using Python: A Computational Problem-Solving Focus*. Wiley, 2012, 612 p.
16. Millman K.J., Aivazis M. Python for scientists and engineers. *Computing in Science and Engineering*, 2011, vol. 13, no. 2, pp. 9–13. doi: 10.1109/MCSE.2011.36
17. Rostanski M., Grochla K., Seman A. Evaluation of highly available and fault-tolerant middleware clustered architectures using RabbitMQ. *Proc. 2014 Federated Conference on Computer Science and Information Systems*. Warsaw, Poland, 2014, vol. 2, pp. 879–884. doi: 10.15439/2014F48

Авторы

Самохин Никита Юрьевич – инженер, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, ORCID ID: 0000-0002-4634-8809, samon@vuztc.ru

Хоружников Сергей Эдуардович – кандидат физико-математических наук, доцент, начальник департамента информационно-коммуникационной инфраструктуры, заведующий кафедрой, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, Scopus ID: 6508306061 ORCID ID: 0000-0002-3982-9575, xse@vuztc.ru

Трубникова Виктория Михайловна – старший преподаватель, Университет ИТМО, Санкт-Петербург, 197101, Российская Федерация, ORCID ID: 0000-0001-9549-8451, vmtrubnikova@corp.ifmo.ru

Ахмедзянова Регина Радиковна – инженер, ООО «Хостланд», Санкт-Петербург, 197342, Российская Федерация, ORCID ID: 0000-0003-2999-5446, regina.ahm@mail.ru

Булыкина Анастасия Борисовна – инженер, ООО «Хостланд», Санкт-Петербург, 197342, Российская Федерация, ORCID ID: 0000-0003-1180-0671, a.bulykina@mail.ru

Authors

Nikita Yu. Samokhin – engineer, ITMO University, Saint Petersburg, 197101, Russian Federation, ORCID ID: 0000-0002-4634-8809, samon@vuztc.ru

Sergey E. Khoruzhnikov – PhD, Associate Professor, Department Head, Head of Chair, ITMO University, Saint Petersburg, 197101, Russian Federation, Scopus ID: 6508306061 ORCID ID: 0000-0002-3982-9575, xse@vuztc.ru

Victoria M. Trubnikova – Senior lecturer, ITMO University, Saint Petersburg, 197101, Russian Federation, ORCID ID: 0000-0001-9549-8451, vmtrubnikova@corp.ifmo.ru

Regina R. Akhmedzyanova – engineer, "Hostland", Saint Petersburg, 197342, Russian Federation, ORCID ID: 0000-0003-2999-5446, regina.ahm@mail.ru

Anastasia B. Bulykina – engineer, "Hostland", Saint Petersburg, 197342, Russian Federation, ORCID ID: 0000-0003-1180-0671, a.bulykina@mail.ru