DOI: https://doi.org/10.15276/hait.07.2024.28
UDC 004.942

Software tools for organizing cloud computing in psychophysiological research based on eye-tracking data

Vitaliy D. Pavlenko[1)]
ORCID: https://orcid.org/0000-0002-5655-4171; pavlenko_vitalij@ukr.net
Andrii S. Ilutsa[1)]
ORCID: https://orcid.org/0009-0007-3445-8481; ilutsa.a.s@op.edu.ua
Vadym I. Gidulian[1)]
ORCID: http://orcid.org/0009-0003-1227-130X; vadim.gidulyan@gmail.com
[1)] Odesa Polytechnic National University, 1, Shevchenko Ave. Odesa, 65044, Ukraine

ABSTRACT

The architecture and web version of the software complex have been developed, which significantly expands the diagnostic capabilities of model-oriented information technologies for the assessment of the neurophysiological state. The complex provides cross-platform cloud computing, increases the productivity and efficiency of scientific research, using methods of non-parametric identification of the oculomotor system based on eye-tracking data, which is achieved thanks to a new concept of cloud computing organization. Cloud computing technology has been further developed thanks to the proposed concept that combines the principles of PaaS (Platform as a Service) and SaaS (Software as a Service). The key feature of the complex is the interface builder and the code translation module, which provide flexibility and convenience of working with the complex, allowing you to configure interface elements and connect them with script-code in different languages. Automatic replacement of values in script-code simplifies the adaptation of the complex to various tasks, making it accessible to users with any skill level, which is especially valuable for science and education. In addition, the important feature of this software complex is its undemanding hardware on the client side thanks to the use of cloud computing, and its modular structure, which allows it to be easily scaled. Compared to other similar services, the complex has several advantages: it provides effective work in research and educational areas, supports several programming languages for improving algorithms, and also allows the use of ready-made identification methods through specially developed GUI interfaces. In addition, it offers social capabilities and a high level of abstraction that optimizes the research process.

Keywords: Web-application; cloud services; cloud computing; PaaS; SaaS; eye-tracking technology; neurophysiological research, code translation

***For citation*: Pavlenko V. D., lutsa A. S., Gidulian V. I. "Software tools for organizing cloud computing in psychophysiological research based on eye-tracking data". *Herald of Advanced Information Technology*. 2024; Vol.7 No.4: 393–404. DOI: https://doi.org/10.15276/hait.07.2024.28**

1. INTRODUCTION

Examining the relationship between oculomotor dynamic characteristics and the central nervous system, along with analyzing an individual's psycho-emotional state, contributes to a deeper understanding of brain mechanisms, their disorders, the dynamics of psychophysiological states, as well as processes related to perception, thinking, imagination, and differentiation of personal intentions and attitudes. Eye-tracking technology actively utilized in diagnostic studies of neurophysiological states [1, 2], [3, 4], [5], research on cognitive processes and memory [6, 7], and for monitoring student behavior and learning [8]. Such studies allow for a more profound understanding of both conscious and subconscious aspects of human behavior. Knowledge of eye movements has both theoretical and applied significance, opening new possibilities for studying the characteristics of various professions to enhance work efficiency.The extensive use of hardware for innovative eye-tracking technology in experimental studies of neural processes necessitates the development of specialized software for managing large data sets [9, 10], [11, 12]. There is a demand for reliable and accurate indicators of mental health for both individuals and population groups, as well as substantiated indicators for monitoring data reliability and validity. Emotion recognition technologies make it possible to infer the state of the nervous system and assess it in everyday high-risk situations. By analyzing changes in eye movement trajectories, can be formed specific conclusions about the psychophysiological state of the subjects.

Implementing this technology in scientific research across various institutions and educational establishments is feasible through the use cloud services.

Cloud services have become one of the main driving forces of modern digital transformation. They provide access to computing resources, such as servers, data storage, databases, networks, software analytical tools, and much more via the Internet [13, 14], [15, 16]. These resources can be offered in various models, allowing users to optimize the use of IT resources, reduce costs, and increase efficiency.

Cloud computing offers several different service models [17, 18], [19], each with its own advantages and suited for different needs:

- **Software as a Service (SaaS)**. This is the most common model, which allows users to access software via the Internet. Users can work with applications without the need to install or maintain them on local devices.
- **Platform as a Service (PaaS)**. PaaS provides an environment for developing, testing, and deploying applications, reducing the complexity of managing infrastructure.
- **Infrastructure as a Service (IaaS)**. IaaS provides virtual computing resources, such as servers, data storage, networks, and other fundamental computing elements that users can use to create and manage their own IT systems.
- **Function as a Service (FaaS).** FaaS is a subset of PaaS, where users can run functions or pieces of code in response to specific events without worrying about server management.

Cloud technologies designed to consolidate IT infrastructures, outsource IT resources, and pool computing resources based on shared-access servers, storage, networks, and software, providing dynamic scalability, flexibility, low costs, and accessibility.

Cloud computing is one of the most important and rapidly evolving technologies of today, transforming various industries [20, 21]. It offers scalability, flexibility in different scientific fields, and economic benefits for businesses by reducing IT infrastructure costs, increasing productivity, and efficiently managing data. Despite existing challenges, such as security and privacy concerns [22, 23], [24, 25], cloud computing continues to evolve, integrating with other advanced technologies and offering new opportunities for innovation. Also considered the integration of cloud computing with other technologies, such as the Internet of Things (IoT) [26, 27], [28, 29] and Blockchain [30, 31] for improve the security of processes. These innovations open up new opportunities for businesses, providing them with competitive advantages in the market. Given this, it is important to continue research in this area to fully harness the potential of cloud technologies in the future.

Thus, cloud computing has become an indispensable tool in modern business and science, opening new horizons for the development and improvement of processes in various fields.

Currently, services such as Jupyter [32] and Google Colab [33] are used to support cloud computing. These are more like interactive notebooks than platforms for conducting experiments and working with data processing results. They serve as editors and execution environments for a single programming language and do not provide the ability to work on projects with code written in multiple programming languages simultaneously. Additionally, they lack the capability to interact with already implemented interfaces for cloud computing in neuro-physiological research based on eye-tracking data.

There is a need for a new concept of organizing cloud computing in neuro-physiological research based on eye-tracking data and for a software suite/platform that will support two types of cloud computing, PaaS and SaaS, simultaneously. This will enable effective work in research and educational fields with code written in multiple programming languages, such as Python and JavaScript, to improve algorithms, as well as with previously implemented identification methods in the form of GUI interfaces. Additionally, considering the large volume of data in projects on this platform and their possible interdependence, a social component will be essential to facilitate data sharing between researchers and projects, enhancing the productivity of scientific research, which is lacking in similar existing solutions.

2. OBJECTIVE OF THE WORK

The objective of this work is to develop the architecture and functional web version of a software platform that enhances the diagnostic capabilities of model-oriented information technology tools for neurophysiological state assessment. This platform ensures cross-platform application compatibility and improves the productivity and effectiveness of scientific research by utilizing non-parametric identification methods of the oculomotor system based on eye-tracking data, enabled through the proposed new concept of cloud computing organization.

3. RESEARCH RESULTS

A new concept for cloud computing organization has proposed and developed (Fig. 1), which integrates two types of platform-user interaction: PaaS (Platform as a Service) and SaaS (Software as a Service). This approach enables cloud computing be organized in a way that allows researchers to work on projects through both an embedded interface and at the software code level, enhancing the platform's capabilities through an integrated specialized editor.

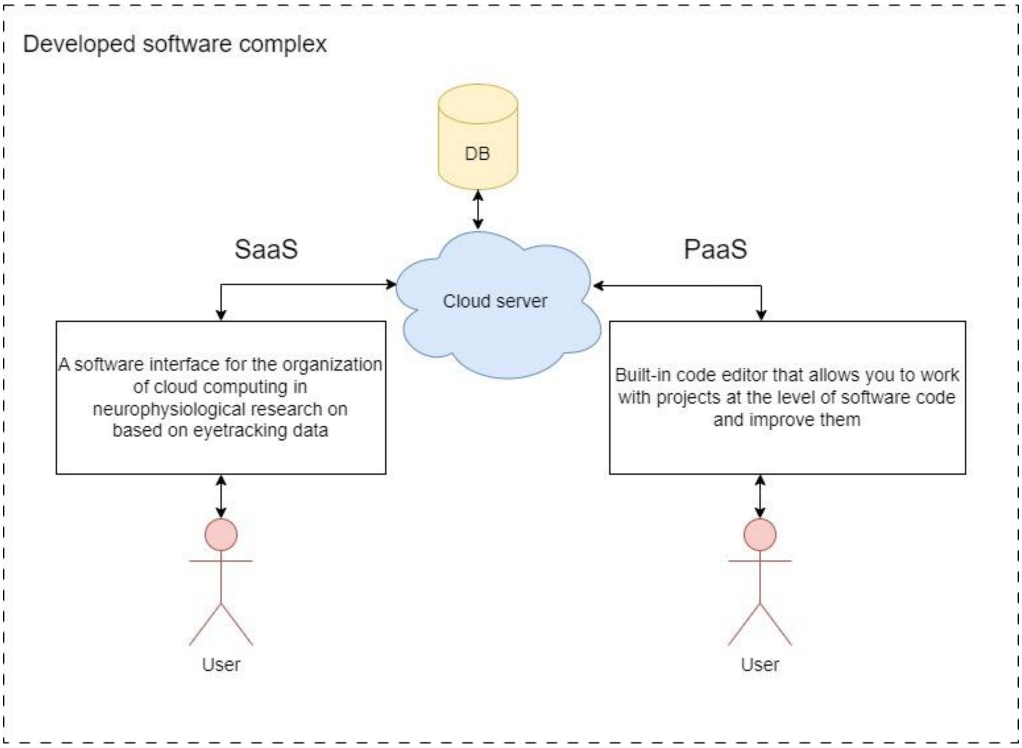


Fig. 1. **Concept of cloud computing organization**
Source: **compiled by the authors**

A software platform has been developed to automate and optimize research processes based on the identification of nonlinear dynamic systems, allowing the addition of new methods through an integrated code editor functioning as a PaaS service. This platform also enables users to edit and execute code for any identification method (provided the code meets documentation requirements), add experiment parameters, perform computations on the server via integrated script code, collect results on the client side in a browser, and save them. Additionally, the platform provides specialized preparation of oculomotor system experimental data obtained from an eye tracker for further processing in nonlinear dynamic identification procedures, facilitated by a specially designed interface that operates as SaaS service.

The platform comprises multiple modules and nodes that interact with each other, illustrated by the functional workflow scheme in Fig. 2:

- **Independent Server Component**: includes all computational modules, core logic, and data processing. Each module operates autonomously and can be scaled independently.
- **Client Component**: provides the user interface and client logic through a cross-platform web interface in the form of an SPA (single-page application).

From a functional perspective, the software product consists of four main components:

- **Server**: manages all technical processes, including receiving requests, providing responses, generating tasks for agents, and storing information in the database.
- **Web Interface**: provides users with access to the service and the ability to work on projects.
- **Agents**: responsible for executing assigned tasks and interpreting code.
- **Database**: designed to store all information related to the platform and projects.

The Fig. 2 illustrates all possible interactions between the structural elements of the service, specifically:

A1, A3 – The client sends a task to the server;

B1 – The server sends a task to the agent;

B2 – The agent sends results back to the server;

A2, A4 – The server sends results/data to the client;

C1 – The server stores information in the database;

A4 – The server retrieves information from the database.

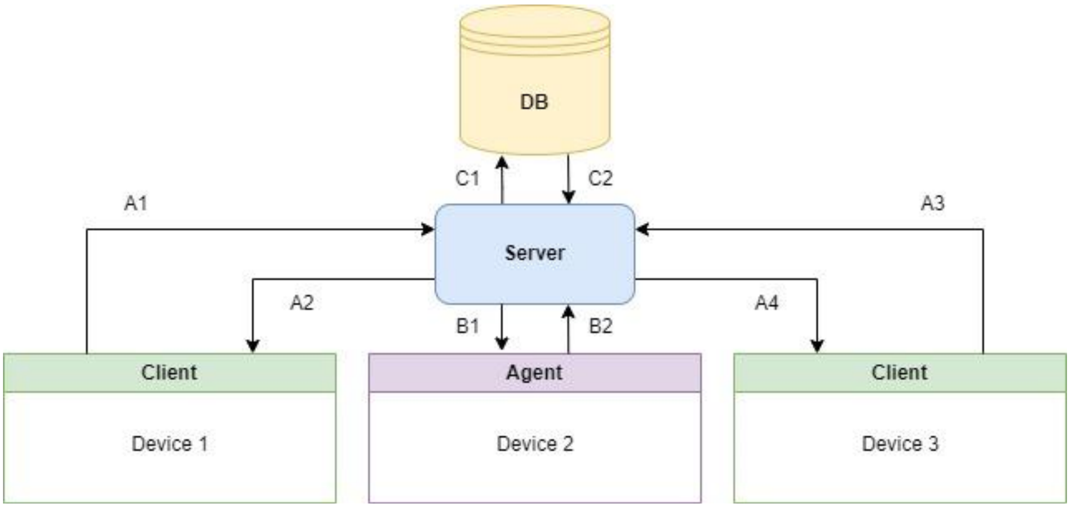


Fig. 2. **Functional workflow scheme of the complex**
Source: **compiled by the authors**

The shown in Fig. 3 structure of cloud computing organization enables independent development and scaling of both components of the platform, creating opportunities for the further addition of mobile and proprietary applications across any platform (Windows, Linux, Android, MacOS, iOS). By centralizing all logic and data processing on the server side, an API interface has been created, allowing external applications to integrate the functionalities of the developed software platform. This provides additional opportunities for project scalability.

Firebase API – this part of the service handles all client requests and sends requests to the database or service used as the database (Firebase library). The independence of this part of the system allows for easy integration of clients from other platforms (Windows, MacOS, Linux, Android, iOS, etc.).

Code Execution Environment – an isolated part of the software system that acts as a program for executing Python and JavaScript code, on which the scripts of all identification methods are run. Results are then sent to the client side for further review and, if necessary, stored in the database. Python and JavaScript virtual machines are used for code interpretation.

One of the main processes is the execution of project script code. This process and the process of providing results to the client can be described by the following pseudocode:

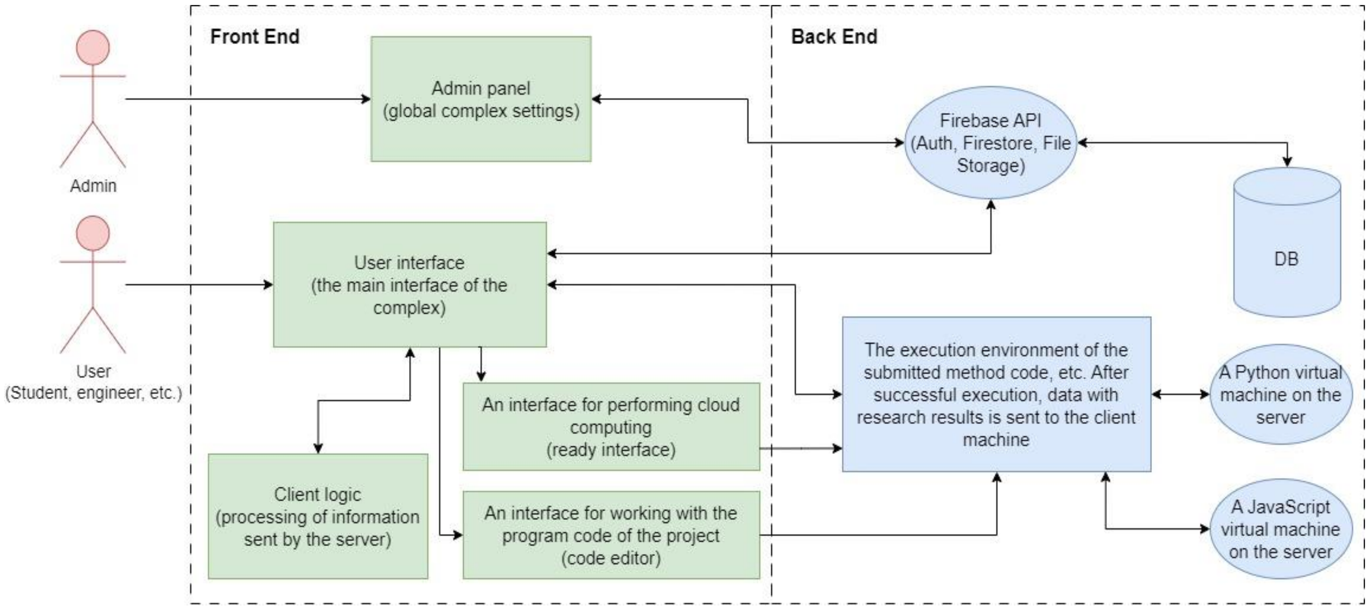


Fig. 3. **The structure of the software complex**
Source: **compiled by the authors**

```
algorithm execute_code_on_server is
input: project
if (is_running) return
if (!project.code) return
out := null
err := null
is_running := true
out, err := request('POST /',
project.code, project.files)
is_running := false
if (err)
show_errors(err)
return
graphs = parse(out)
show_results(graphs)
```

The process that takes place on the server to provide results can be described by the following pseudocode:

```
algorithm server is
on POST / do
uid := generate_unique_id()
project_path := get_project_path(uid)
create_project_dir(project_path)
main_path := project_path + "main.ext"
save_file(request.code, main_path)
for each file in request.files do
save_file(file, project_path)
out, err := execute(main_path)
return uid, out, err
```

Firebase Library – a library of components that, through its API, allows working with the Firebase Firestore NoSQL database for storing user data. It also supports Firebase Auth for user authentication and access control, as well as Firebase Storage for file storage.

Firebase Auth – a module for user authentication in the system. It supports social media vendors like Facebook, GitHub, Twitter, Google, and Google Play Games, and includes a user management system for authenticating users with email and password stored in Firebase.

Firebase Storage provides reliable file uploads and downloads for Firebase applications, regardless of network quality. Developers can use it to store images, audio, video, or other user-generated content. Firebase Storage is supported by Google Cloud Storage.

The process of working with the software platform to obtain results consists of the following steps, illustrated in Fig. 4.

As shown in the figure, the modules are independent, allowing for more efficient scaling without affecting each other. If needed, alternative clients for different platforms or tasks can be connected to the server modules. At the same time, the server can be expanded with additional modules without requiring changes to existing clients.

4. INTERFACE CONSTRUCTOR AND CODE TRANSLATION

The interface constructor and code translator is a tool that enables users to customize interaction parameters with script code without needing deep technical expertise. As shown in Fig. 5, it supports: creating fields for different data types, such as single or multiple values, making it versatile for various scenarios; easy integration of created elements into the source script code using special constructs.

Users can create a new field, assign it a name, add a brief description, and select a data type. This functionality allows for not only customizing variable values but also creating dynamic arrays or parameter tables, which can be used during algorithm execution

The constructor is implemented based on declarative programming, which focuses on describing the final outcome rather than step-by-step code execution. In this setup, the user simply specifies the data they wish to obtain or modify, while the system delegates processing to the code translator. The declarative approach conceals low-level implementation details, lowering the entry barrier and allowing users to concentrate on the task's essence. For example, in the first image, users can add interface elements that automatically integrate into the script code, offering complete control over parameter behavior without manual code modification.

Computational linguistics plays a critical role in developing the constructor, helping to identify and process the syntactic structures users employ to bind interface elements with code. This enables the interface to interpret complex variables and parameters, such as $A3$ or A, and link them to relevant code segments. This design avoids hardcoding and allows dynamic variability, making it easy for users to modify values without risking disruption to the program's logic.

A distinctive feature of the constructor is its use of specific syntactic constructions to connect interface elements with code. As illustrated in Fig.5, variable $A3$ links to the script code, substituted by a user-defined value. This mechanism allows the system to automatically substitute values during compilation or execution, ensuring flexibility and ease in configuring parameters. This capability enables the code to be adapted to new tasks effortlessly, retaining readability and simplicity.

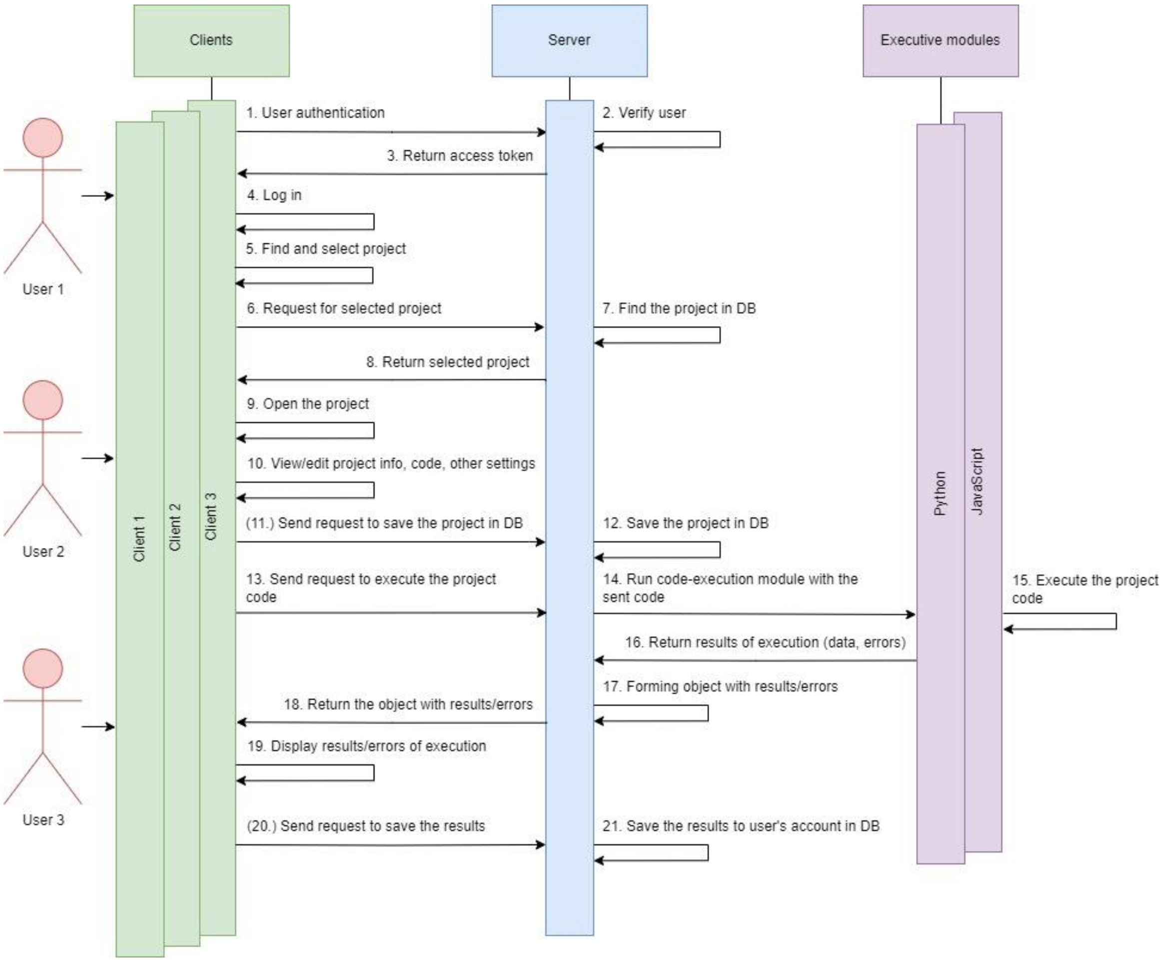


Fig. 4. **The process of working with the software complex**
Source: **compiled by the authors**

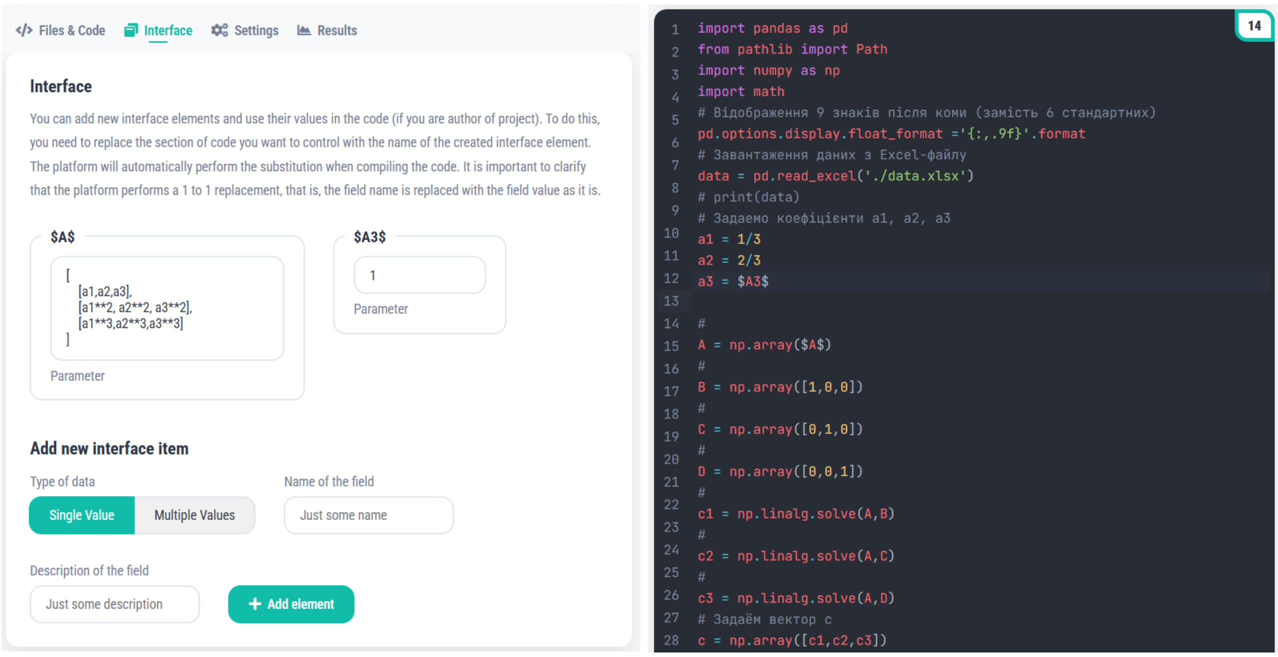


Fig. 5. **Example of interface constructor and in-code integration**
Source: **compiled by the authors**

The code translator developed for the system is a fundamental component, responsible for interpreting user-defined declarative descriptions and converting them into executable code. During translation, the translator identifies all markers and replace them with corresponding values or arrays specified in the interface. This design guarantees continuous and reliable interaction between the interface and executable code, enabling users to update parameters on the fly without rewriting the entire code. The translator's architecture is designed to support complex data types and multi-level dependencies, making it exceptionally flexible and adaptable.

To illustrate the underlying logic of the code translator and its integration with interface elements, the following pseudo-code provides an abstract overview of the translation and execution process, highlighting the key functions and their interactions without delving into specific implementation details:

```
initialize_storage()

add_element(marker, value):
store_marker_and_value(marker, value)

initialize_translator(elements):
for each element in elements:
add_element(element.marker, element.value)

translate_code(script, elements):
initialize_translator(elements)
for each line in script:
for each marker in stored_elements:
if marker is found in line:
substitute(marker with stored_value)
return script

execute_translated_code(script, elements):
translated_script = translate_code(script,
elements)
execute(translated_script)

main():
script = load_script()
elements = load_interface_elements()
execute_translated_code(script, elements)

main()
```

Explanation of Abstracted Pseudo-code:

- **initialize_storage:** Sets up the storage mechanism for markers and values, abstracting away the details of storage implementation.
- **add_element:** Adds each marker and associated value to storage, encapsulating how they're stored.
- **initialize_translator:** Initializes the translator by adding all elements from the interface to storage, preparing for translation.
- **translate_code:** The core translation function. It first initializes the translator with the given elements. Then, it iterates over each line of the script, checking for markers and substituting each found marker with its stored value.
- **execute_translated_code:** Handles the full process of translating the code and then executing it, abstracting away specific execution details.
- **main:** The main control function that loads the script and interface elements, then initiates the translation and execution process.

In conclusion, the interface constructor, coupled with the code translator, provides a powerful tool for creating adaptive systems tailored to specific user tasks. This approach speeds up the development process, making it more intuitive and flexible, especially for non-technical users. It opens up new possibilities for using the tool in both academic research and scientific research where rapid code adaptation to changing conditions is required.

5. ISOLATION OF USER SCRIPT EXECUTION IN A CLOUD COMPUTING ENVIRONMENT

In modern cloud computing it is important to execute user-submitted code securely and efficiently. This is particularly relevant in environments that support multiple programming languages where users expect real-time code execution through a web-based interface. Given the inherently untrusted nature of user-submitted code, ensuring robust security, effective isolation, and resource management is crucial. In this context, Docker [34] containerization is a good solution to address these challenges while also simplifies the setup and management of diverse execution environments.

Security. Executing untrusted user code introduces significant security risks, especially when such code runs on shared infrastructure. A critical concern is the potential for malicious code to access sensitive data, either belonging to other users or the system itself. Docker containers offer a layer of security by isolating each execution environment. Each container functions as a sandboxed environment with its own filesystem, processes, and network stack, preventing user code from interacting with the host system or other containers. By restricting the capabilities of the containers, the attack surface is minimized, ensuring that even if a user attempts to exploit the system, any damage is contained within the isolated environment.

Isolation. Isolation is a fundamental requirement for executing user code in a multi-tenant environment. Docker's containerization technology provides process and resource isolation. As shown in Fig. 6, each container runs in its own isolated environment, ensuring that the code executed within it does not interfere with or gain access to other user’s containers or the host system. This isolation is achieved through namespaces and groups, core features of the Linux kernel that Docker leverages. Namespaces provide isolated views of system resources, such as process trees, network interfaces, and filesystems, while groups manage and limit resource usage for each container. The isolated environment within a Docker container allows for the concurrent execution of multiple user scripts without the risk of one user's script affecting another's. This is crucial for maintaining the integrity and reliability of the service and providing a consistent experience for all users. In the event that a user's script behaves unpredictably, it will be contained within its designated Docker container, preventing it from impacting the broader system or other users' workloads.

Resource Management. Managing resource consumption is another critical aspect of executing user code in a cloud environment. Docker containers enable precise control over resource allocation, such as CPU usage, memory limits, and disk I/O. This is particularly important when dealing with potentially resource-intensive scripts that could monopolize server resources and degrade performance for other users. By setting limits on CPU and memory usage, administrators can ensure that no single container consumes more than its fair share of resources, thereby maintaining system stability and performance.

Flexibility in supporting multiple programming languages. One of the significant advantages of using Docker for executing user code is the ease with which diverse execution environments can be set up and managed. Docker allows for the creation of containers tailored to specific programming languages, each with its own set of dependencies, libraries, and runtime configurations. This approach ensures that user code is executed in a consistent and reproducible environment.

Moreover, by designing these containers with a unified interface for handling code execution, the introduction of new programming languages becomes a straightforward process. Once a container template is established for a particular language, adding support for additional languages requires minimal effort.

Utilizing Docker for executing user code in a cloud computing environment offers substantial advantages in terms of security, isolation, resource management, and flexibility.

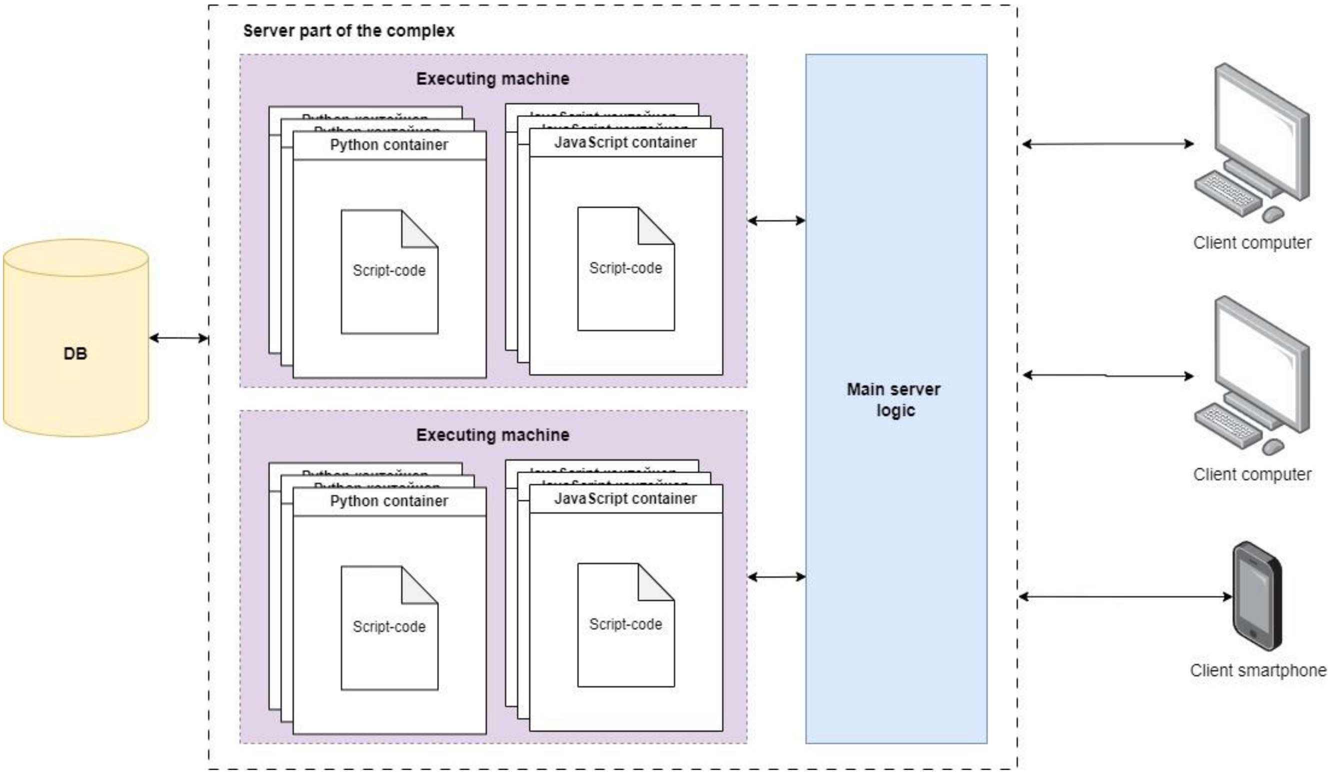


Fig. 6. **The principle of isolation of script code execution**
Source: **compiled by the authors**

Docker's containerization technology not only provides robust security and isolation but also simplifies the setup and management of diverse execution environments for efficient signal processing and real-time data analysis. The ability to enforce fine-grained resource controls and easily extend support to new programming languages makes Docker an ideal solution for securely and efficiently running untrusted code in cloud-based web applications. These features collectively contribute to a more resilient and scalable system, capable of meeting the demands of modern cloud computing.

6. SCIENTIFIC NOVELTY

For the first time, a method for diagnosing the neurophysiological state of an individual has been proposed based on model-oriented intelligent information technology and experimental eye-tracking data, supported by cloud computing. This approach significantly enhances the productivity and efficiency of scientific research, allowing multiple researchers to participate in the diagnostic process, regardless of their geographical location.

Cloud computing technology has been further developed through a new concept of cloud service organization that operates simultaneously on the principles of Platform as a Service (PaaS) and Software as a Service (SaaS). This approach provides advantages over known counterparts, simplifies research and educational processes, and ensures the platform's minimal hardware requirements and cross-platform compatibility.

CONCLUSIONS

An architecture and software platform have been developed within a new cloud computing concept, enhancing the diagnostic capabilities of model-oriented information technology for neurophysiological state assessment. The platform ensures cross-platform compatibility for cloud computing processes, increasing the productivity and effectiveness of scientific research through non-parametric identification methods of the oculomotor system based on eye-tracking data.

A primary feature of the platform is its combination of PaaS and SaaS principles, alongside a modular structure that simplifies scalability. The platform enables effective work in research and educational fields with code in various programming languages, such as Python and JavaScript, for algorithm refinement and supports ready-made identification methods through specially designed GUI interfaces. It allows users to access the platform as a cloud service on any device.

One of the primary components of the platform is an interface constructor, enabling users to easily create and customize interface elements that link with code segments through a specially developed translator. This translator automatically replace values specified in the interface into the appropriate sections of the code, making the configuration and interaction with the software intuitive and user-friendly. This feature allows users to adapt the platform to new research needs without extensive programming knowledge.

Additionally, a significant advantage of this software platform is its low hardware requirements on the client side due to cloud computing technology. It surpasses services like Project Jupyter, Google Colab by enabling work with popular programming languages and pre-built methods through customized GUI interfaces, a high level of abstraction using the developed interface constructor, and enhanced social features, optimizing the research process and offering new opportunities to users. The platform supports script execution for computation methods in Python and JavaScript.

The software development utilized tools such as JavaScript for general development, HTML and CSS for interface creation, the Vue.js framework [35], Python for implementing nonlinear dynamic identification methods, and Node.js for client-server interaction.

REFERENCES

1. Opwonya, J., Doan, D. N. T., Kim, S. G. et al. "Saccadic eye movement in mild cognitive impairment and alzheimer's disease: A systematic review and eta-analysis". *Neuropsychol Rev*. 2022; 32: 193–227. DOI: https://doi.org/10.1007/s11065-021-09495-3.

2. Jansson, D., Rosén, O. & Medvedev A. "Parametric and nonparametric analysis of eyetracking data by anomaly detection". *IEEE Transaction Control System Technology*. 2015; 23: 1578–1586. DOI: https://doi.org 10.1109/TSCT.2014.2364958.

3. Bro, V. & Medvedev, A. "Continuous and discrete Volterra-Laguerre models with delay for modeling of smooth pursuit eye movements". *IEEE Transactions on Biomedical Engineering.* 2023; 70 (1): 97–104.

4. Lanata, L., Sebastian, L., Di Gruttola F., Di Modica S., Scilingo, E.P. & Greco1, A. "Nonlinear analysis of eye-tracking information for motor imagery assessments". *Frontiers in Neuroscience*. 2020; 13: 1–11. DOI: https://doi.org 10.3389/fnins.2019.01431.

5. Pavlenko, V., Milosz, M. & Dzienkowski, M. "Identification of the oculo-motor system based on the Volterra model using eye tracking technology". *4th Int. Conf. on Applied Physics, Simulation and Computing (APSAC')*. *Journal of Physics: Conference Series*. Rome, Italy. 2020; 1603: 1–8.

6. Keehn, B., Monahan, P., Enneking, B., et al. "Eye-Tracking biomarkers and autism diagnosis in primary care". *JAMA Netw Open*. 2024; 7 (5): 1–14. DOI: https://doi.org/10.1001/jamanetworkopen. 2024.11190.

7. Pavlenko, V. & Shamanina, T. "Eye Tracking in the Study of Cognitive Processes". *The 6th International Conference on Computational Linguistics and Intelligent Systems (CoLInS')*. 2022. DOI: https://doi.org/10.30888/2709-2267.2022-13-01-013.

8. Weiss, K., Kolbe, M., Lohmeyer, Q. & Meboldt, M. "Measuring teamwork for training in healthcare using eye tracking and pose estimation". Front. Psychol; 14: 1169940. 2023 p. 1–12. DOI: https://doi.org/10.3389/fpsyg.2023.1169940

9. Pavlenko, V., Shamanina, T., Chori, V. "Eyetracking technology and its application in neuroscience". *Proceedings of the the 12th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*. Dortmund, Germany. 2023; 1: 187–193.

10. Pavlenko, V., Shamanina, T. & Chori V. "Biometric identification based on the multidimensional transient functions of the human oculo-motor system". *Journal of Physics Conference Series*. 2022; 2162: 012024. DOI: https://doi.org/10.1088/1742-6596/2162/1/012024.

11. Pavlenko, V. D., Shamanina, T. V. & Chori, V. V. "Estimation psychophysiological state via integral nonlinear model of the oculo-motor system". *Applied Aspects of Information Technology*. 2023; 6 (2): 117–129. DOI: https://doi.org/10.15276/aait.06.2023.8.

12. Pavlenko, V. D., Shamanina, T. V. & Chori, V. V. "Nonlinear dynamics identification of the oculo-motor system based on eye tracking data". *International Journal of Circuits, Systems and Signal Processing* 2021; 15: 569–577. (E-ISSN: 1998- 4464). DOI: https://doi.org/10.46300/9106.2021.15.63.

13. Saraswat, M. & Tripathi, R. C. "Cloud computing: Analysis of top 5 CSPs in SaaS, PaaS and IaaS platforms". *9th International Conference System Modeling and Advancement in Research Trends (SMART.)* 2020. p. 300–305

14. Kollipara, P. "An overview on cloud computing". *International Journal of Research in Engineering, Science and Management*. 2021; 4 (8): 35–37.

15. Fatima, E., Sumra, I. A. & Naveed, R. "A comprehensive survey on security threats and challenges in cloud computing models (SaaS, PaaS and IaaS)". *Journal of Computing & Biomedical Informatics*. 2024; 7 (01): 537–544.

16. Singh, A., Sharma, S., Kumar, S. R. & Yadav, S. A. "Overview of PaaS and SaaS and its application in cloud computing". *International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)*. 2016. p. 172–176.

17. Mohammed, C. M. & Zeebaree, S. R. "Sufficient comparison among cloud computing services: IaaS, PaaS, and SaaS". *International Journal of Science and Business*. 2021; 5 (2): 17–30.

18. Wulf, F., Lindner, T., Strahringer, S. & Westner, M. "IaaS, PaaS, or SaaS? The why of cloud computing delivery model selection. Vignettes on the post-adoption of cloud computing". *Proceedings of the 54th Hawaii International Conference on System Sciences*. 2021. p. 6285–6294.

19. Nadeem, F. "Evaluating and ranking cloud IaaS, PaaS and SaaS models based on functional and non-functional key performance indicators". *IEEE Access*. 2022; 10: 63245–63257. DOI: https://doi.org/10.1109/ACCESS.2022.3182688.

20. Yoo, S. K. & Kim, B. Y. "The effective factors of cloud computing adoption success in organization". *The Journal of Asian Finance, Economics and Business*. 2019; 6 (1): 217–229.

21. Faridi, F., Sarwar, H., Ahtisham, M. & Jamal, K. "Cloud computing approaches in health care". *Materials Today Proceedings*. 2022; 51 (1): 1217–1223.

22. Parast, F. K., Sindhav, C., Nikam, S., Yekta, H. I., Kent, K. B. & Hakak, S. "Cloud computing security: A survey of service-based models". *Computers & Security*. 2022; 114: 102580.

23. Mehboob, T., Sumra, I. A. & Shahzadi, I. "Detection of DDoS/DoS attack methodologies in cloud computing network". A Survey. *Journal of Computing & Biomedical Informatics.* 2024. – Available from: https://jcbi.org/index.php/Main/article/view/191.

24. Arunkumar, M. & Ashokkumar, K. "A review on cloud computing security challenges, attacks and its countermeasures". *In AIP Conference Proceedings*. 2024; 3037 (1): 020047. DOI: https://doi.org/10.1063/5.0196063.

25. Damaraju, A. "Cloud security challenges and solutions in the era of digital transformation". *International Journal of Advanced Engineering Technologies and Innovations*. 2024; 1 (3): 387–413.

26. Tyagi, H. & Kumar, R. "Cloud computing for IoT". *Internet of Things (IoT) Concepts and Applications.* 2020. p. 25–41.

27. Sadeeq, M. M., Abdulkareem, N. M., Zeebaree, S. R., Ahmed, D. M., Sami, A. S. & Zebari, R. R. "IoT and Cloud computing issues, challenges and opportunities: A review". *Qubahan Academic Journal.* 2021; 1 (2): 1–7.

28. Kaur, C. "The cloud computing and internet of things (IoT)". *International Journal of Scientific Research in Science, Engineering and Technology.* 2020; 7 (1): 19–22.

29. Rani, S., Bhambri, P., Kataria, A., Khang, A., Sivaraman, A. K. (Eds.). "Big data, cloud computing and IoT: Tools and applications". *CRC Press*. 2023.

30. Habib, G., Sharma, S., Ibrahim, S., Ahmad, I., Qureshi, S. & Ishfaq, M. "Blockchain technology: benefits, challenges, applications, and integration of blockchain technology with cloud computing". *Future Internet.* 2022; 14 (11): 341.

31. Zou, J., He, D., Zeadally, S., Kumar, N., Wang, H. & Choo, K. R. "Integrated blockchain and cloud computing systems: A systematic survey, solutions, and challenges". *ACM Computing Surveys (CSUR).* 2021; 54 (8): 1–36.

32. Project Jupyter. – Available from: https://jupyter.org.

33. Google Colub. – Available from: https://colab.research.google.com.

34. Docker. – Available from: https://www.docker.com.

35. Vue.js framework. – Available from: https://vuejs.org.

Conflicts of Interest: The authors declare that they have no conflict of interest regarding this study, including financial, personal, authorship or other, which could influence the research and its results presented in this article

Received 10.09.2024
Received after revision 15.11.2024
Accepted 21.11.2024

DOI: https://doi.org/10.15276/hait.07.2024.28
УДК 004.942

Програмні засоби організації хмарних обчислень у психофізіологічних дослідженнях на основі даних айтрекінгу

Павленко Віталій Данилович[1)]
ORCID: https://orcid.org/0000-0002-5655-4171; pavlenko_vitalij@ukr.net. Scopus Author ID: 54401442600
Ілуца Андрій Сергійович[1)]
ORCID: https://orcid.org/0009-0007-3445-8481; ilutsa.a.s@op.edu.ua
Гідулян Вадим Ігорович[1)]
ORCID: http://orcid.org/0009-0003-1227-130X; vadim.gidulyan@gmail.com
[1)] Національний університет «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна

АНОТАЦІЯ

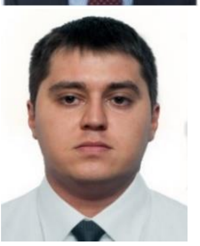
Розроблено архітектуру та веб-версію програмного комплексу, який значно розширює діагностичні можливості модельно-орієнтованої інформаційної технології для оцінки нейрофізіологічного стану. Комплекс забезпечує кросплатформеність хмарних обчислень, підвищує продуктивність і ефективність наукових досліджень. Комплекс використовує методи непараметричної ідентифікації окорухової системи на основі даних айтрекінгу, що досягається завдяки новій концепції організації хмарних обчислень. Технологія хмарних обчислень отримала подальший розвиток завдяки запропонованій концепції, що поєднує принципи PaaS (Platform as a Service) і SaaS (Software as a Service). Ключовою складовою комплексу є конструктор інтерфейсів і модуль трансляції коду, які забезпечують гнучкість і зручність роботи з комплексом, дозволяючи налаштовувати елементи інтерфейсу і з'єднувати їх з скрипт-кодом на різних мовах. Автоматична заміна значень у скрипт-коді спрощує адаптацію комплексу до різних завдань, роблячи його доступним для користувачів будь-якого рівня кваліфікації, що особливо цінно для науки та освіти. Також, важливою особливістю цього програмного комплексу є його невимогливість до апаратного забезпечення на стороні клієнта завдяки використанню хмарних обчислень та модульна структура, що дозволяє легко його масштабувати. Порівняно з іншими подібними сервісами комплекс має кілька переваг: він забезпечує ефективну роботу в дослідницьких і навчальних напрямках та підтримує декілька мов програмування для вдосконалення алгоритмів. Комплекс також дозволяє використовувати готові методи ідентифікації через спеціально розроблені GUI-інтерфейси. Крім того, він пропонує соціальні можливості та високий рівень абстракції, що дозволяє оптимізувати дослідницький процес.

Ключові слова: Web-застосунок; хмарні сервіси; хмарні обчислення; PaaS; SaaS; технологія айтрекінгу; психофізіологічні дослідження; трансляція коду

ABOUT THE AUTHORS

Vitaliy D. Pavlenko - Doctor of Engineering Sciences, Professor, Department “Computerized Systems and Software Technology” Odesa Polytechnic National University, 1, Shevchenko Ave. Odesa, 65044, Ukraine
ORCID: https://orcid.org/0000-0002-5655-4171; pavlenko_vitalij@ukr.net. Scopus Author ID: 54401442600
***Research field*:** Math modelling and Simulation; Volterra models; nonlinear dynamic identification; intelligent information technology; machine learning; technical and medical diagnostics; neuroscience; biometric identification; innovation eye-tracking technology

Павленко Віталій Данилович - доктор технічних наук, професор, професор кафедри «Комп’ютеризовані системи та програмні технології» Національного університету «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна



Andrii S. Ilutsa - PhD student, Department “Computerized Systems and Software Technology” Odessa Polytechnic National University, 1, Shevchenko Ave. Odesa, 65044, Ukraine
ORCID: https://orcid.org/0009-0007-3445-8481; ilutsa.a.s@op.edu.ua
***Research field*:** Software engineering; UI/UX design; technical and medical diagnostics; cloud computing

Ілуца Андрій Сергійович - магістр**,** аспірант кафедри «Комп’ютеризовані системи та програмні технології» Національного університету «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна

Vadim I. Gidulian - Getter, Department “Computerized Systems and Software Technology”. Odesa Polytechnic National University, 1, Shevchenko Ave. Odesa, 65044, Ukraine
ORCID: http://orcid.org/0009-0003-1227-130X; vadim.gidulyan@gmail.com
***Research field*:** Software engineering; databases; data analysis; cloud computing

Гідулян Вадим Ігорович - здобувач кафедри «Комп’ютеризовані системи та програмні технології» Національного університету «Одеська політехніка», пр. Шевченка, 1. Одеса, 65044, Україна

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)