Models and Tools for Designing Adaptive User Interfaces of Intelligent Systems Ecosystem

Mikhail Sadouski and Alexandra Zhmyrko and Vitali Tsishchanka
Belarusian State University of
Informatics and Radioelectronics
Minsk, Belarus
Email: sadovski@bsuir.by, aleksashazh@gmail.com, vitalik.tsishchanka@gmail.com

***Abstract*—This article discusses the principles of automation of user actions with the help of personal assistants, models of adaptive user interfaces of Ecosystem of intelligent systems, as well as means of individual editing of models of such interfaces.**

***Keywords*—user interface of the OSTIS Ecosystem, OSTIS Ecosystem, personal assistant, Web LLM Agent, knowledge base editing**

I. Introduction

Modern technology has given rise to countless services and systems designed to address diverse needs. However, this abundance has created a major challenge: users must learn the intricacies of each system's unique *user interface*, making the interaction cumbersome and time-consuming. Many *user interfaces* are unintuitive, requiring specialized skills just to perform basic tasks, forcing users to constantly adapt rather than focus on their goals.

This is where a *personal assistant* becomes essential. By serving as a unified intermediary, it eliminates the need to master every *user interface* individually. Instead of struggling with inconsistent designs, users can rely on the *personal assistant* to seamlessly navigate systems, translating their intent into the right actions across platforms.

The solution, therefore, lies in prioritizing *personal assistants* that simplify and standardize interactions with the *Ecosystem of intelligent systems*. By acting as a single point of control, they improve usability, reduce learning curves, and dramatically improve efficiency. The future of seamless technology is not just a better user experience: it is an assistant that bridges the gap between users and the *Ecosystem of intelligent systems*.

The purpose of this article is to analyze existing solutions to this problem and an approach to eliminating this problem using the *OSTIS Technology*.

II. State of the art

Currently, in order to solve the problem of usability of various information services and unification of *user interfaces*, the following is used:

- *personal assistants* that simplify the processes of user interaction with various systems;
- standards of protocols and *application interfaces* that simplify the integration and interaction of various systems with each other.

Digital *personal assistants* are programs based on artificial intelligence technologies that help users perform everyday tasks, such as scheduling, managing contacts, searching for information, reminding about important events, etc. [1]–[4].

Nowadays many companies try to develop their own *personal assistants*, and the first company that managed to integrate such an agent into their operating systems was Apple when they introduced Siri [5] in 2010. Soon after, many other companies implemented assistance in the same area in order to help people perform ordinary everyday actions (Microsoft Cortana [6], Google Now [7], LG Voice Mate [8], [9]).

The *user interface* of the *personal assistant* must represent the system as a single unified set of back-end task assistants, enabling the user to conduct a dialog in which it is easy to switch between these domains. This involves getting user input commands either as text or speech and processing the natural language input to understand it [10].

While a *personal assistant* streamlines user interactions, it does not inherently resolve the deeper issue of cross-system compatibility. Standardized protocols and *Application Programming Interfaces* can fight this difficulty but a complete solution requires building an *Ecosystem* of semantically interoperable systems where services seamlessly understand and integrate with one another.

The study [9], introduces LISSA, a *personal assistant* designed to support students in a Virtual Education Space (VES). This VES acts like a smart *Ecosystem*, similar to an Internet of Things (IoT) network, where independent intelligent tools work together while adapting to user needs. By integrating LISSA into this system, students get effective learning help, and the assistant can easily grow to handle more tasks in the future.

Several years of active development of *large language models(LLMs)* have provided developers with new opportunities to approach problem solving. It is achieved by the ability of *LLM* to suggest a solution for a task that is not very precisely described.

Some of the most popular branches of works in the *LLM* environment and other neural network methods and tools are recognition of the elements of the user interface, automation of user interface actions and generation of user interfaces by a text description.

For frontend software development, commercial and open-source tools such as Vercel's V0, Imagica and OpenV0 have attracted considerable interest using the capabilities of *LLM* [11]. These applications adeptly transform user-provided textual or visual prompts into concrete, well-structured high-fidelity *user interfaces*, along with their associated frontend code, providing substantial support to designers and developers.

It has become possible for systems to have a set of primitive actions, and the system itself will decide in what order to perform these primitive actions. Examples of such primitive actions might be:

- to open a web page;
- to click a button;
- to fill in a text input field;
- to read some string from a page;
- to recognize an image.

Thus the performance of some action in the system can be broken down into a sequence of similarly primitive actions, and the task of deciding in what order and with what arguments these actions will be performed can be delegated to the *LLM*.

Entities that perform primitive user interface actions are called *Web LLM Agents* or *UI LLM Agents* [12].

A *Web LLM agent* uses the set of states of the agent in which it can be and some *context* in which the agent operates. The *context* includes the set of allowed actions, the history of actions performed, results of performed actions, actions that were not performed due to their lack of helpfulness, and the agent's environment, that is, a description of what is happening on the screen. At each step, the agent uses the instructions, the environment (screen), the set of allowed actions, and the history of executed actions to select the most appropriate action for the situation, which may or may not change the agent's state. Action selection and action execution occur until the agent reaches the final state or until the maximum number of actions given to the agent to complete the task has been performed.

All allowed states are manually selected and described in detail by the developer. This is done to make it easier for agents to use the information about the state in which they are currently in. Observations of the environment that will come from the outside are thought out, and an example of an observation coming to the agent is provided. In the example of such an observation, the parameters to be changed are replaced by placeholders, so that the real values with which the agent interacts can be substituted for them. The final goal and a detailed description of how to act in each state are also given.

In order to prevent the *LLM* from suggesting actions that are currently unavailable, a set of allowed actions is defined for each state. This may resemble a finite-state machine in which transitions do not necessarily change the state of the machine. An alternative or complement to this approach is to add the actions of all buttons on the page to the set of allowed actions. This would increase the agent's capabilities, but there is a risk that it would be more difficult for the agent to choose which action to take at the moment.

Examples of agent's states for the task of buying a product that meets user's requirements is shown on Figure 1.

It is possible to use several *LLMs*, so that one model reasoned and described what is on the page, what should be achieved and what are the options in current state, and the other *LLM* made a decision on what action to perform based on this reasoning.

[13] describes an approach on how to adapt the LLM UI Agent to various *user interfaces* with recognition of elements of the webpage in order to adapt to unseen websites and domains. The authors review a method for learning with fine-tuning and human demonstration for solving this task. Similar solution parses regions of *user interfaces* screenshots into structured elements, which can be used as an input for the *LLM* for suggesting next action [14].

III. Proposed approach

The analysis shows that users should not need to search through countless services to find what they need. Instead, a network of compatible, interconnected services should work invisibly in the background. For this to succeed, all resources and tools must seamlessly understand each other (semantic compatibility). The user's *personal assistant* and not the user should handle selecting the right service for each task.

Thus, when implementing digital *personal assistants*, it is necessary to ensure their scalability and adaptability to user needs. This means that the system should be able to automatically adapt to changes in user behavior, taking into account their preferences, work characteristics, and other factors.

The *OSTIS Technology* allows creating semantically compatible systems (*ostis-systems*) that are able to process user requests and problems, taking into account their context and meaning. This is achieved through the use of semantic networks that allow describing knowledge and the connections between it. Also, the *OSTIS Technology* provides scalability and flexibility of the system, which allows it to adapt to changes in user behavior and needs [15].

The *OSTIS Technology* provides a universal language for the semantic representation (encoding) of information

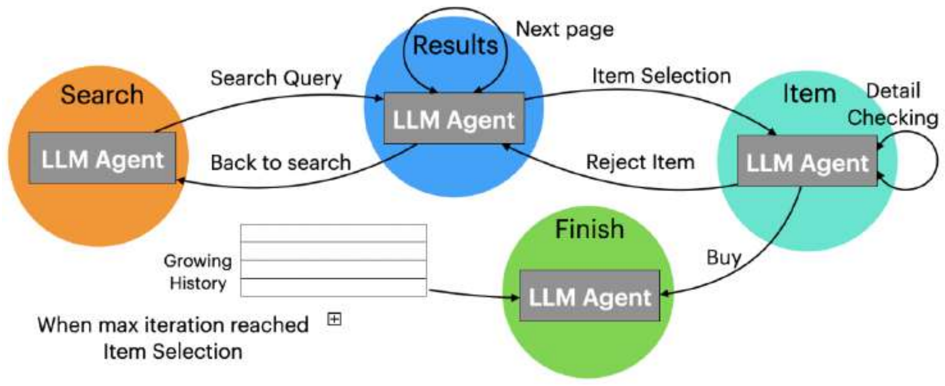

Figure 1. Web LLM Agent's states for the task of buying a product

in the memory of *intelligent computer systems*, called the *SC-code*. Texts of the *SC-code* (sc-texts) are unified semantic networks with a basic set-theoretic interpretation, which allows solving the problem of compatibility of various knowledge types. The elements of such semantic networks are called *sc-elements* (*sc-nodes* and *sc-connectors*, which, in turn, depending on orientation, can be *sc-arcs* or *sc-edges*). The *Alphabet of the SC-code* consists of five main elements, on the basis of which *SC-code* constructions of any complexity are built, including more specific types of *sc-elements* (for example, new concepts). The memory that stores *SC-code* constructions is called semantic memory, or *sc-memory* [16].

The architecture of each *ostis-system* includes a platform for interpreting semantic models of *ostis-systems*, as well as a *semantic model of the ostis-system* described using the *SC-code* (*sc-model of the ostis-system*). In turn, the *sc-model of the ostis-system* includes the *sc-model of the knowledge base*, *sc-model of the interface*, and *sc-model of the problem solver*. The principles of the design and structure of *knowledge bases* and *problem solvers* are discussed in more detail in [17] and [18], respectively. The principles of the *sc-model of the user interface* were described in the articles [19], [20], and [21], on which this article is based.

Within the *OSTIS Technology*, the concept of the *OSTIS Ecosystem* is introduced [22].

The *OSTIS Ecosystem* is a socio-technical network, which is a collective of interacting:

- *ostis-systems* themselves;
- users of the specified *ostis-systems* (both end-users and developers);
- some *computer systems* that are not *ostis-systems* (they can be used as additional information resources or services).

The objectives of the *OSTIS Ecosystem* are:

- rapid implementation of all agreed changes in *ostis-systems*;
- permanent maintenance of a high-level of mutual understanding between all the systems that are part of the *OSTIS Ecosystem*, as well as all their users;
- corporate solution of various complex problems requiring the coordination of several (most often a priori unknown) *ostis-systems* and possibly some users.

Within the *OSTIS Ecosystem*, the concept of a *personal ostis-assistant* is specified. A *personal ostis-assistant* is the *ostis-system*, which is a *personal assistant* of the user within the *OSTIS Ecosystem*. Such a system provides opportunities:

- to analyze user activity and form recommendations for its optimization;
- to adapt to the mood of the user, their personal qualities, the general environment, the problems that the user most often solves;
- to permanently train the assistant in the process of solving new problems, while learnability is potentially unlimited;
- to conduct a dialog with the user in natural language, including in speech form;
- to conduct a dialog with various systems within the Ecosystem;
- to provide answers to questions of various classes, while asking counter-questions in case the system does not understand something;
- to independently receive information from the entire environment, and not just from the user (in text or speech form).

At the same time, the system can both analyze available information sources (for example, on the Internet) and analyze the physical world surrounding it, for example, surrounding objects or the appearance of the user.

Advantages of the *personal ostis-assistant*:

- the user does not need to store different information in different forms in different places: all information is stored in a single *knowledge base* compactly and without duplication;

- thanks to unlimited learnability, assistants can potentially automate almost any activity, not just the most routine one;
- thanks to the *knowledge base*, its structuring, and the means of searching for information in the *knowledge base*, the user can get more accurate information more quickly.

Since the user interaction with the *OSTIS Ecosystem* occurs only through a *personal ostis-assistant*, the *user interface of the OSTIS Ecosystem* for the user is the *user interface* of their *personal ostis-assistant*. Such an *interface* should be *adaptive*, *intelligent*, and *multimodal*. The structure of such an *interface* was proposed in the work [21].

The model of the *adaptive user interface* occupies a significant place in such an interface. The model of the *adaptive user interface* consists of the model of the *knowledge base* of the *adaptive user interface* and the model of the *problem solver* of the *adaptive user interface*. In its turn the model of the *knowledge base* of the *adaptive user interface* is a union of the model of user interfaces and components of user interfaces, the model of the user context, the model of the external information constructions and external languages, the model of incoming and outgoing messages of intelligent systems and the model that contains methodology of user interface design. The model of the user context contains the model of users, the model of user actions, the model of the environment and the model of devices.

The *ostis system* design is based on a <u>component-based approach</u>. That means that any system can use prepared library of components that will reduce the time that the creation of a system requires and enlarge the functionality of the system and the efficiency of the design. That components are created with the use of the *OSTIS Technology* which makes them easy to integrate in any *ostis system*.

All components are stored as fragments of the *knowledge base* of the *ostis system* which makes them ready to be used by decision making processes described earlier.

A unique feature of the *OSTIS Technology* is ensuring the <u>compatibility</u> for the *components of ostis-system knowledge bases*, *ostis-system problem solvers*, and *ostis-system interfaces* due to a <u>single unified formal basis</u>. Thus, a *user interface component* for its work must usually include not only a description of its visual characteristics in the *knowledge base* but also *components of the problem solver* (for example, *sc-agents*), as well as the necessary fragments of some *subject domain*.

As it was shown in previous works [23], it is already possible to integrate *LLM* interactions into the *OSTIS Technology*, and due to that it is possible to create context for UI LLM Agents, manage static and dynamic prompts creation and collect purposes and abilities of *user interfaces* and *problem solvers* of the systems that create *Ecosystem*.

The advantages of the *OSTIS Technology*, and in particular the transparency it provides, will make it much more efficient for web agents to operate while handling user needs. Action planning makes better results in cases when each component of each system in thoroughly specified.

Another helpful trait of *ostis systems* is that each *user interface* element is described in the *knowledge base*, making it possible for agents to know how to interact with it and to know what it can be used for. That helps with adding an ability to explain the reasoning of choosing specific button to press or any other interface action.

For navigating between different *ostis systems* in the *Ecosystem* in order to find the one that can solve the required task, the summary of the components in each system can be used by agents. That summary, combined with other context parts, such as current state, previous actions and results of those actions and description of states, may be successfully retrieved, compiled as a part of a dynamic prompt and used as input for *LLM*.

One of the key aspects of having large *knowledge bases* is having proper and convenient tools for visualising and editing parts of that knowledge graph. Current *Ecosystem* of *ostis systems* supports two languages for knowledge visualisation: SCn and SCg. The first one has syntax and semantic rules based on indents and special symbols. In comparison based on that principle SCg does not use any additional rules for knowledge visualisation. It has just nodes and connectors between nodes and connectors that are directly represent sc-nodes and sc-connectors in *knowledge base* and it should take a couple of minutes to teach someone how to write and read SCg code.

In order to generate new parts of the *knowledge base*, personal tools for *knowledge base* editing should be accessed by personal assistant.

For easier integration of new visualisation formats for *personal assistants* that will meet any users needs, it was decided to create a pipeline of agents that take some sc-structure and desired format and translate that sc-structure into the provided format.

Problem solver for translating an sc-structure
$\Rightarrow$ *decomposition of abstract sc-agent**:
  {• *Abstract sc-agent for dividing a structure*
    $\Rightarrow$ *note**:
      [The structure is divided into connectivity components]
    $\Rightarrow$ *implementation**:
      C++ language
  • *Abstract sc-agent for main key sc-element selection*
    $\Rightarrow$ *note**:
      [For each connectivity component

and each structure the main key sc-element is selected (it may be that initially the main key element is absent or there are several of them, but after that step each connectivity component is having only one main key sc-element)]

⇒ *implementation**:
C++ language

- *Abstract sc-agent for classifying a structure*
  ⇒ *note**:
  [Each connectivity component is classified to be a logical rule, weighed graph, scp-program etc.]
  ⇒ *implementation**:
  C++ language
- *Abstract sc-agent for sc-elements ordering*
  ⇒ *note**:
  [For each connectivity component, semantic ordering of all key elements takes place (the order in which this connectivity component will be perceived when reading)]
  ⇒ *implementation**:
  C++ language
- *Abstract sc-agent for translating into visualisation*
  ⇒ *note**:
  [Translation into the given visualization language takes place: SCn or SCg]
  ⇒ *implementation**:
  C++ language
- *Abstract sc-agent for translating into an external format*
  ⇒ *note**:
  [The translation to the external format of the target user interface interpreter]
  ⇒ *implementation**:
  C++ language

}

The result of applying this sc-agents pipeline is shown on Figure 2. This visualisation is aimed at eliminating intersections of connectors and it is easier to perceive fragments of the *knowledge base*.

Agents are always able to get description of elements placed on the screen because proposed approach suggests usage of an agent-oriented model for interpreting the semantic model of *adaptive user interfaces*. That model is able to generate *user interface* based on the task the user is trying to complete and adapts to the user's environment and actions.

IV. Conclusion

The article analyses the problem of the existence of a large number of tools that are similar in functionality but different in appearance, which makes it difficult for users to comfortably meet their needs. Modern approaches use neural network methods such as recognizing *user interface* elements by screenshot, *Web LLM Agents* and generating interfaces by description. This problem can be solved by utilizing a standard for *user interface* design and using that standard to create *adaptive user interfaces*.

It is proposed to use the *OSTIS Technology Standard* to reduce the impact of heterogeneous interfaces. A structured knowledge base created using the proposed model of *adaptive user interfaces* will simplify the process of creating *user interfaces*.

Suggested approach assumes the use of the *OSTIS Technology*, which includes the *OSTIS Ecosystem* and *personal ostis-assistants* to ensure effective and comfortable user interaction with the *Ecosystem*. *OSTIS Technology* ensures that actions that system perform are transparent and it is possible to explain the need of each action performed.

Within the proposed approach, the *user interface of the OSTIS Ecosystem* is considered as the *user interface of a personal ostis-assistant*, since the user interacts with the *Ecosystem* only through their *personal assistant*. The principles of the *user interface of the OSTIS Ecosystem* were described, with the main one being the usage of the *component approach* to design and the possibility for a *personal assistant* to use any *user interface component* within the *OSTIS Ecosystem*. Models for the *user interface of the OSTIS Ecosystem* were proposed and tool for that interface is described, using the example of a personal tool for *knowledge base* editing.

Acknowledgment

The work was carried out with partial financial support of the BRFFR (contract № F24MV-011 from 15.04.2024).

References

[1] C. Meurisch, M.-D. Ionescu, B. Schmidt, and M. Mühlhäuser, "Reference model of next-generation digital personal assistant: Integrating proactive behavior," in *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, ser. UbiComp '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 149–152. [Online]. Available: https://doi.org/10.1145/3123024.3123145

[2] C. Meurisch, C. A. Mihale-Wilson, A. Hawlitschek, F. Giger, F. Müller, O. Hinz, and M. Mühlhäuser, "Exploring user expectations of proactive ai systems," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 4, no. 4, Dec. 2020. [Online]. Available: https://doi.org/10.1145/3432193

[3] J. Paay, J. Kjeldskov, E. Papachristos, K. M. Hansen, T. Jørgensen, and K. L. Overgaard, "Can digital personal assistants persuade people to exercise?" *Behaviour & Information Technology*, vol. 41, no. 2, pp. 416–432, 2022.

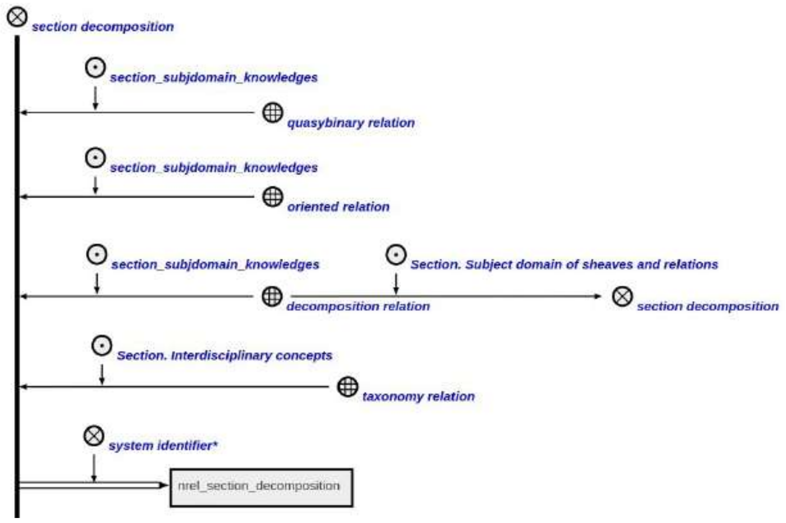
section decomposition
section_subjdomain_knowledges
quasybinary relation
section_subjdomain_knowledges
oriented relation
section_subjdomain_knowledges
Section. Subject domain of sheaves and relations
decomposition relation
section decomposition
Section. Interdisciplinary concepts
taxonomy relation
system identifier*
nrel_section_decomposition

Figure 2. SCg visualised using sc-agents pipeline

[4] A. Akbar, "Proactivity in intelligent personal assistants: A simulation-based approach," in *Multi-Agent Systems*, D. Baumeister and J. Rothe, Eds. Springer International Publishing, 2022, pp. 423–426.

[5] (2023, Mar) Siri web page. [Online]. Available: https://www.apple.com/siri/

[6] (2023, Mar) Microsoft Cortana web page. [Online]. Available: https://www.microsoft.com/en-us/cortana

[7] (2023, Mar) Google Now web page. [Online]. Available: https://www.digitaltrends.com/mobile/what-is-google-now/

[8] (2023, Mar) LGvoice mate web page. [Online]. Available: https://www.lg.com/us/mobile-phones/VS985/Userguide/388.html

[9] J. Todorov, V. Valkanov, and I. Popchev, "Intelligent personal assistant for aiding students," 10 2017.

[10] S. Ahmad and M. Imran, "Asr based intelligent personal assistant," 03 2019.

[11] M. Yuan, J. Chen, Z. Xing, A. Quigley, Y. Luo, T. Luo, G. Mohammadi, Q. Lu, and L. Zhu, "Designrepair: Dual-stream design guideline-aware frontend repair with large language models," 2024. [Online]. Available: https://arxiv.org/abs/2411.01606

[12] K. Ma, H. Zhang, H. Wang, X. Pan, W. Yu, and D. Yu, "Laser: Llm agent with state-space exploration for web navigation," 2024. [Online]. Available: https://arxiv.org/abs/2309.08172

[13] G. Verma, R. Kaur, N. Srishankar, Z. Zeng, T. Balch, and M. Veloso, "Adaptagent: Adapting multimodal web agents with few-shot learning from human demonstrations," 11 2024.

[14] Y. Lu, J. Yang, Y. Shen, and A. Awadallah, "Omniparser for pure vision based gui agent," 2024. [Online]. Available: https://arxiv.org/abs/2408.00203

[15] V. Golenkov, N. Gulyakina, and D. Shunkevich, *Otkrytaya tekhnologiya ontologicheskogo proektirovaniya, proizvodstva i ekspluatatsii semanticheski sovmestimykh gibridnykh intellektual'nykh komp'yuternykh sistem [Open technology of ontological design, production and operation of semantically compatible hybrid intelligent computer systems].* Minsk: Bestprint, 2021, (In Russ.).

[16] V. Golenkov, N. Gulyakina, I. Davydenko, and D. Shunkevich, "Semanticheskie tekhnologii proektirovaniya intellektual'nyh sistem i semanticheskie associativnye komp'yutery [Semantic technologies of intelligent systems design and semantic associative computers]," *Otkrytye semanticheskie tehnologii proektirovanija intellektual'nyh sistem [Open semantic technologies for intelligent systems]*, pp. 42–50, 2019.

[17] I. Davydenko, "Semantic models, method and tools of knowledge bases coordinated development based on reusable components," in *Otkrytye semanticheskie tehnologii proektirovanija intellektual'nyh sistem [Open semantic technologies for intelligent systems]*, V. Golenkov, Ed., BSUIR. Minsk , BSUIR, 2018, pp. 99–118.

[18] D. Shunkevich, "Agentno-orientirovannye reshateli zadach intellektual'nyh sistem [Agent-oriented models, method and tools of compatible problem solvers development for intelligent systems]," in *Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems]*, V. Golenkov, Ed. BSUIR, Minsk, 2018, pp. 119–132, (In Russ.).

[19] A. Boriskin, M. Sadouski, and D. Koronchik, "Ontology-based design of intelligent systems user interface," *Otkrytye semanticheskie tekhnologii proektirovaniya intellektual'nykh system [Open semantic technologies for intelligent systems]*, pp. 95–106, 2017.

[20] M. Sadouski, "Semantic-based design of an adaptive user interface," in *Open Semantic Technologies for Intelligent Systems*. Springer, 2022, pp. 165–191.

[21] ——, "The structure of next-generation intelligent computer system interfaces," *Open semantic technologies for intelligent systems*, pp. 199–208, 2022.

[22] A. Zagorskiy, "Principles for implementing the ecosystem of next-generation intelligent computer systems," *Open semantic technologies for intelligent systems*, pp. 347–356, 2022.

[23] K. Bantsevich, M. Kovalev, V. Tsishchanka, N. Malinovskaya, and A. Andrushevich, "Integration of large language models with knowledge bases of intelligent systems," *Open Semantic Technologies for Intelligent Systems*, pp. 213–218, 2023.

МОДЕЛИ И СРЕДСТВА ПРОЕКТИРОВАНИЯ АДАПТИВНЫХ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ ЭКОСИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Садовский М. Е., Жмырко А. В.,
Тищенко В. Н.

В статье рассматриваются принципы автоматизации действий пользователя при помощи испльзования персональных ассистентов, модели адаптивных пользовательских интерфейсов Экосистемы интеллектуальных систем, а также средства индивидуального редактирования моделей таких интерфейсов.

Received 31.03.2025