УДК 004.75

Y. Vovk

TECHNOLOGY OF BOT CREATION FOR WIDE CLASS OF LOGIC APPROACH-BASED APPLICATIONS

Abstract: The article proposes information technology for computer-aided design and implementation of bots on the base of logical approach. The general approach to creation of bots with use of a complex of logical and linguistic models is described. The general scheme of work of the web-oriented system with use of a chat bot is developed. The model of logic of actions is presented and the theorem proving method what can be used for the formal description of process of creation of bots is described. An example of using a logical model to create a bot is described.

Keywords: Bot creation technology, action logic, state logic, chat bot, output system.

Introduction.

Human activity - work, study or free time - is regulated by template-algorithmic processes. For state institutions, business enterprises, companies it is waiting in queues, writing official letters, transitions from one office to another. In the field of recreation it is search for people who are ready to spend their free time in a certain way. In educational institutions it is, on one hand, notifying students and their parents about changes, events that occur or will happen in the institution, on the other hand - providing answers to the students’ and their parents’ questions about these changes, events, issuing various certificates. Such processes regulate work, but introduce delays.

So, a need for technologies for managing virtual human interaction in solving a wide range of problems emerges. The tools of interaction are electronic document management systems, e-mail, educational services for student-teacher interaction, specialized services for finding people by interests, location, etc., various messengers and social networks.

However, use of such information society tools has disadvantages that cause negative consequences for system owners and their users. The system owner pays a relatively small but fixed rent during the period of use of the system within the license agreement with the developer or one-time but significant sums for the development of such a system. In this case, regardless of the method of acquisition, the owner of the system is not deprived of the cost of maintaining this system in working order by adding the necessary content, scaling in case of increasing the number of users, etc.

In the case of the user, the disadvantages are related to the system use. Because modern user has an access to online management systems for almost every component of his life and each of these systems has its own interface, which does not always correspond to the UX familiar to user, as well as its own authorization method, login and password (if using a single password on all systems the user violates the rules of safe use of web systems), there is a need for regular decentralized review of information on each area of life within the system or set of systems, memorizing a large number of passwords and features of each system. At least this leads to extra time, and in general we can talk about a number of disadvantages with negative consequences for the user.

In general, the use of decentralized management systems for each individual component of the social component of the user is time-consuming for him and resource-consuming for the system owner. Therefore, a problem of centralized management of all components of user activity in social networks emerges.

The development of modern technologies and the experience of using solutions based on them show that the problem described above can be partially solved by means of rapid automated creation and use of broad-class chatbots. This solution would be better supplemented with a system integrating the user's work with all messengers, social networks and information systems of the information society in terms of content, graphics and features of the owner, but this will be the subject of the next article of the author.

Formulation of the task of bot creation technology development

It is necessary to develop a general approach, mathematical models and methods as the basis of information technology of a wide class of chatbots creation. In essence, we are talking about solving traditional problems of coordination in the "employee - employee", "customer - dispatcher", "student - employee of the dean's office", "student - teacher" schemes with a new solution aimed at creating special programs - bots that automate human functions related to informing, ordering and fulfilling orders, the implementation of multi-step processes, individual stages of which depend on various circumstances, events and characteristics of the participants by using the capabilities of messengers of social networks. The general approach, the specified models and methods should be aimed at fast creation of the programs realizing a choice and realization of the scheme of interaction (service) depending on type and individual features of set of possible variants of their arrangement with subcircuits, thus being deprived of shortcomings of the algorithmic approach associated with the need to reprogram the bot when new user characteristics and change of implemented subcircuits or the emergence of new subcircuits. This technology must meet a high level of requirements, especially its following properties: flexibility, scalability, accessibility to the end user without special knowledge and skills, reliability, convenience and ease of use, distribution, use of remote resources, load adaptation, fast and simple integration with information systems.

Publications analysis

Instant messaging systems are widely used by individual users and human communities around the world. In fact, today they have evolved from simple means of communication between people into means of obtaining information and an incredibly powerful marketing tool. The key part when using them is the ability to communicate with bots. Therefore, this article discusses the basic principles of creating bots based on logical and linguistic models in an approach that captures, accumulates, generalizes and uses a positive user experience to ensure maximum efficiency of its activities in social networks. Therefore, it is important to review developments in this area, analyze scientific publications in order to select the necessary tools for the development of bots, the integration basis for integration into a single system and examples to demonstrate the effectiveness of the proposed solution.

For the last few years, the popularity of instant messaging systems (messengers) has been growing. They have evolved from a means of communication between people into a means of obtaining information and a powerful marketing tool. Bots played a significant role in this. The definition of a chatbot as a computer program that is responsible for user interaction based on auditory or textual methods has became a de facto standard. Many attempts to classify bots exist, from which it is feasible to choose the classification of chatbots by: type of messages (commands) to text and voice; the level of interactivity on the bots with the response to the call of the specified command and the analysis of the message and the selection of the command/commands; type of interaction with the chat server on the poll (Long pooling) and listeners [1].

Most available messengers provide the ability to select the interface of the level of communication between chatbot and the messenger server. The next option is offered at the level of interaction type, where there are two modes of interaction - Long pooling, which regulates the polling of the messenger server via API once in the interval of N milliseconds, where the specification of milliseconds is regulated by a specific API, and event subscription mode. The End point API will be called if there is a change. Regarding the advantages and disadvantages of the regimes in view of the problem of the study, the following well-founded conclusions can be made: The operation of the bot using the first mode creates an additional load on the messenger server, but does not require an SSL certificate of external address and is used mainly in mobile and desktop applications. The operation of the bot using the second mode is designed for interaction of the type "server to server" reduces the load on the messenger server. Therefore, when implementing a solution in accordance with the principles of web-oriented architecture, it seems more appropriate to use the second mode.

This useful tool is interesting for business. Bots have been actively used abroad for several years and are gaining popularity in Ukraine. Thus, a promising niche for IT developers has been created. For potential customers, an important aspect is the perception of the bot commands, a good emphasis on the essence of the user's request and the relevance of his answers [2].

The article proposes a holistic approach to creating bots for the messenger, describes the architecture of information technology for creating bots based on models of mathematical logic, artificial intelligence and linguistics to implement the functions of the bot and its communication with users.

General description of an approach to automated bot creation

The solution being proposed does not cover the full range of tasks related to the centralized management of all components of user activity in social networks by creating web-oriented technologies. However, it is able to simplify user interaction with the system, to gain experience and to create conditions for development of more efficient solutions.

The implementation of such a solution requires building user interaction through a chat within a specific messenger based on a targeted platform for processing requests which implements a complete algorithm of actions in all cases provided by a particular chat. Figure 1 shows a general view of the server architecture of the bot interaction level.

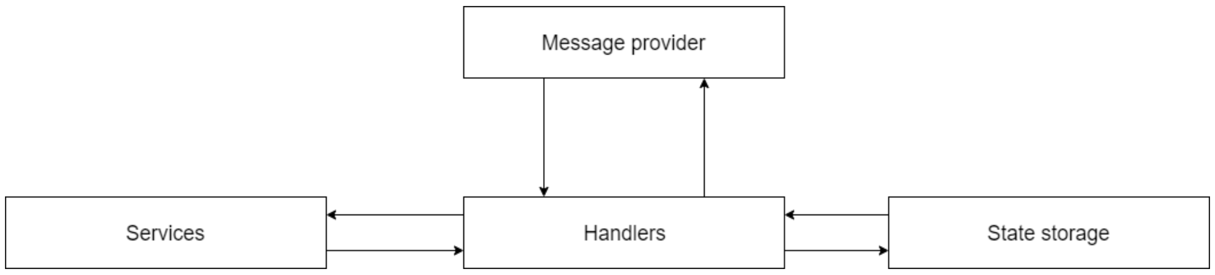

Figure 1. Element of a server architecture on the bot interaction level

Let's consider components of a fragment. *Message provider* component performs the functions of wrapping the message source (messenger). Command handlers (*Handlers*) process messages from the *Message provider,* delegate tasks to services, update the state container, and generate responses to the user. State containet *State storage* stores the parameters of the state of processing user requests. *Services* implement additional logic for processing user requests.

Let’s consider an example of using a chatbot for registration, ordering a taxi, planning the execution of this order and monitoring the implementation of the plan in the system of the international provider of this service. A typical task for chatbots is to order a car. The need to integrate the mechanism of receiving user requests via a chatbot is due to the advantages of user interaction with messengers described above. One of the possible options for specifying the actions, states and requirements for the input data model for the taxi ordering process takes the form of the following sequence: 1. Country selection. 2. Choice of city. 3. Choice of car class. 4. Determining the customer's mobile phone number. 5. Clarification of the time of submission of the car. 6. Clarification of the car delivery address (street, house). States and transitions are shown in Fig. 2. For each state we have a corresponding list of characteristics.

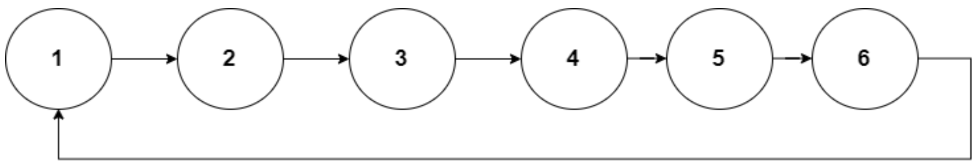

Figure 2. State container: states and transitions

In essence, we have a traditional for control theory model of states and transitions, which turned out to be a convenient form of setting the problem of creating a bot. There are many approaches to the implementation of this model - from a purely algorithmic approach to approaches based on combining problem solving in the space of the system to solve subproblems.

The above example can easily be implemented by the sequence of commands created in the chatbot basing on an algorithmic approach. This easy-to-implement approach is not very convenient for the user and the system owner, as its inherent shortcomings - lack of interactivity and possible changes in the sequence of actions when implementing the order - can lead to significant losses. Thus, adding step 7 to clarify the terms of payment will not change the data, but to rewrite the software module. That is, this mechanism is simple but not flexible.

Approaches that respond to changing states (situations) by taking certain actions and focusing on responding to inquiries about the possibility of achieving certain states (situations) that learn by solving problems have proven to be more flexible and effective. So, the answer to the request is to build an action plan that leads to the target state and which is then implemented in the general case. To do this, each action of the plan included in the system must be described in terms of objects. Achieving the goal in such systems is associated with the derivation of an action plan that provides a solution to the problem.

But the implementation of this approach requires the development of a formal logical model for outlining conditions and solutions and appropriate information technology capable of using the experience and knowledge for automated creation of bots based on logical models and methods of deriving appropriate action plans in the environment of messengers, bots and information systems.

Bot creation technology architecture

Web-oriented architecture of information technology for creating bots based on a logical model is shown in Fig. 3.

Such scheme complements and expands the functionality of the system and makes it user-friendly. We will show that it also provides the system owner with advantages associated with changing the order processing sequence and adding / removing structural elements to the previously implemented sequence of actions. We apply the approach proposed in [4] to the use of templates in the construction of web-applications for the construction of templates in determining the sequence of actions. We modify the requirements for the input data model for the taxi ordering process: 1. Country selection. 2. Choice of city. 3. Choice of car class. 4. Clarification of the mobile phone number (in the absence of the number is available in the messenger). 5. Car feed time. 6. Exact address (street, house). 7. Method of payment. 8. Payment (in case of non-cash payment). 9. Use one of the previous routes. 10. Getting an answer about the status of the order.

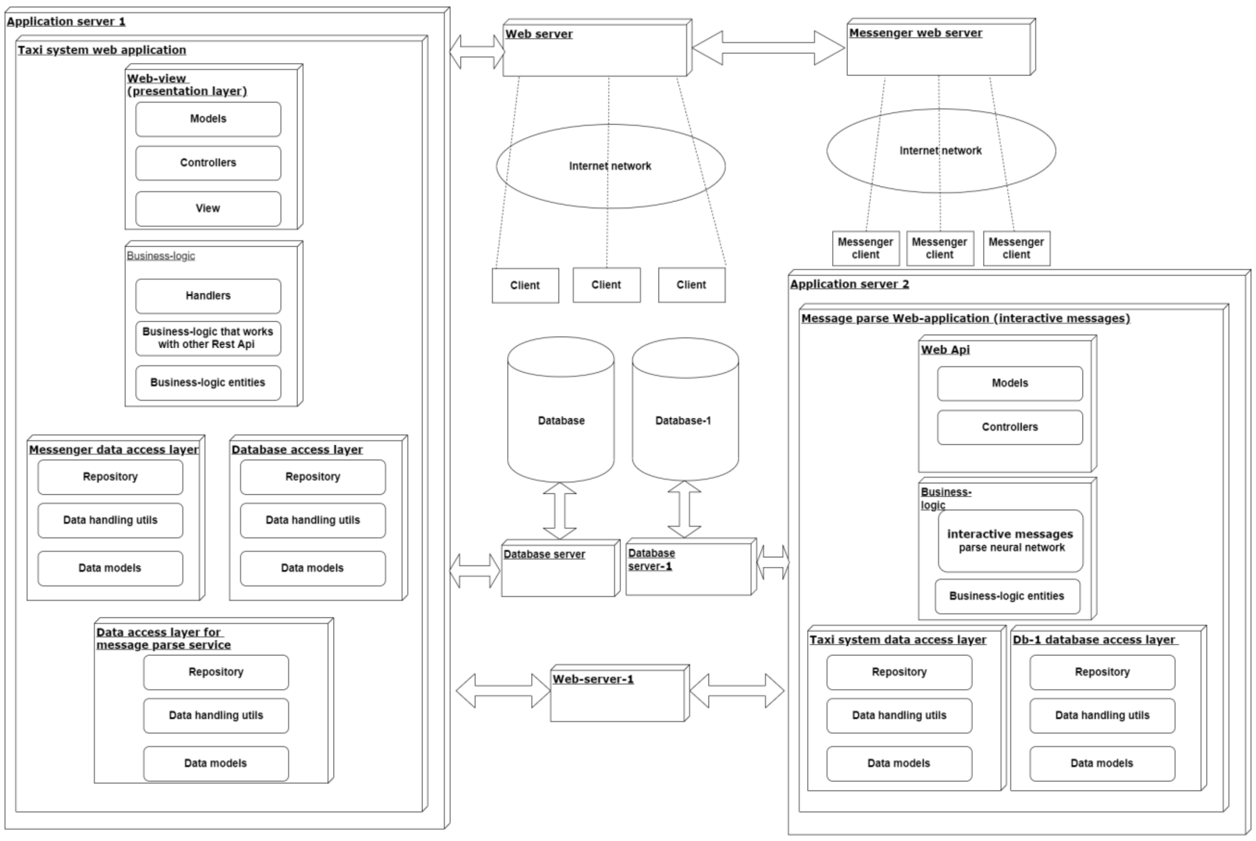

Figure 3. General schema of a web-oriented system using chatbot

$$\begin{tikzcd}
1 \arrow[r] \arrow[ddrrr] & 2 \arrow[rrr] \arrow[dr] & & & 5 \arrow[r] & 6 \arrow[r] & 7 \arrow[drr] \arrow[dr] & & \\
& & 3 \arrow[r] & 4 \arrow[ur] & & & & 8 \arrow[r] & 10 \\
& & & 9 \arrow[urrrrr] & & & & &
\end{tikzcd}$$

Figure 4. Modified state container

We present the logic of states and transitions in accordance with Figs. 4 as a graph of a special kind, the solution of problems on the basis of which is reduced to finding ways using known algorithms. The implementation of such behavior is performed by the available capabilities of the messenger. However, if the requirements for the result are modified and the possibility of renting a car is added, that will require checks on the driver's license, user health, etc., so the algorithm cannot be implemented on the basis of the existing machine state and implemented in modern messengers. Template state logic for chatbots in order to form a result based on the prerequisites and post-conditions of each state by constructing a tree of transitions (method calls) that correctly handle situations from initial to final, user-defined prerequisite and post-condition is needed.

Let’s consider a specific scheme for a clearly defined situation. It is necessary to order a car with the specification of the country, city, class or brand of the car, on lease terms or as a taxi. We have the following prerequisite: the user is authorized, has a license to drive a vehicle. Post-condition: the user wants to rent a car for one day.

The system has a set of methods that implement the sending of the contract, checking the driver's license, city and country, checking the ability to control the user of the vehicle, work with the payment service. The number of methods is not limited, each of them is characterized by a prerequisite (that determines the need to call it) and post-condition (that determines the outcome).

Therefore, the system now operates with a set of states defined by the vertices of the graph, in which the transitions are made on the basis of preconditions and post-conditions, and for each individual user request specified prerequisite (input vertex) and desired result (post-condition). Under this approach, to implement the technology of automated application creation using the chatbot module, it is needed to:

1. Create the template of logic of state construction on the basis of user-entered data (construct a graph, bind conditions to it’s vertices).
2. Create an API Endpoint to receive messages from the messenger.
3. Set in the configuration of the messenger a set of available commands and API Endpoint to receive messenger requests at the time of creating a chatbot.
4. Using the appropriate template, add a Message provider for a messenger at the level of business logic.
5. Receive a message by calling the Message provider from the Endpoint API.
6. Describe the logic of the handler behavior for each command in the format of the result of its execution.
7. Define State storage and create a template for its filling.
8. Connect a third-party server for text message recognition (with further development of an own server) and get a set of commands from it.
9. Generate a class interaction with a third-party API (after 8 is accomplished) using available methods and complement the behavior of handlers at the level of business logic.
10. Supplement calls of handlers with further calls of State storage if a call of the third-party service from point 8 is necessary.

Formal logic system and inference method

Let’s use the formalism proposed in [3], on the basis of which a software system will be built, which will provide a solution to the problem. To create bots with useful behavior for practice, we divide the stages of action planning and implementation of plans. This allows communication at the planning stage with the user in order to quickly and accurately select the next steps, and at the stage of implementation of plans - control of the conditions of previous actions, the appropriate reaction of the environment, redistribution of actions between participants and other adjustments.

The second fundamental step is the division of the planning stage into stages of construction of detailed plans and composition of schemes. At the first stage we make plans. In the second stage, a complex action plan is constructed from simple (template actions) accumulated by the system.

On the one hand, objects (TAgent. TRealObject, TConception, TRelation, TValue, TProblem), situations (states) TSituation), events (TEvent) and actions (TAction) (as well as relevant operations and relationships), and, on the other hand, plans (as well as relevant operations and relationships) are subject to formalization in the logic of action planning, defined in terms of states and transitions, taking into account the allocation of stages of construction of detailed plans and composition of schemes.

The state (situation) characterizes the state (situation) of the control object through its parameters. Implementation is a sequence of states and transitions that fulfills the conditions of initialization, the possibility of transitions, truth and completion.

Formalism is needed to respond to queries that specify the initial and target situation (state) and involve the search or construction of a specific action plan, the implementation of which will ensure the transition of the object of management from the initial situation (state) to the target one.

The main element of the plan is the sequence of actions. In the definition of the plan we enter the constructions of basic actions - empty action “ml”, exit from the plan “exit”, conditional transition, cycle. Given the role of the composition, define the plan as following: 1) any action is a plan; 2) if $\pi 1, \pi 2$ are plans, then $x1, x2$ is a plan too; 3) if $\delta 1$ is an action $\sigma 1$, σn - situations, $\pi 1$, $.., xn$ - plans then case $\delta 1, (\sigma 1, \pi 1), ..., (\sigma n, ... \pi n)$ is a plan; 4) if $x1$ is a plan, and $\sigma 1$ - situation, then while $(\sigma 1, x1)$ is a plan.

To build detailed plans we will use the AP-system, and for the composition of plans - PC-system, proposed in [3]. Let’s consider important for the formalization of the creation of bots features of axiomatizations.

In the AP-system within the framework of clause logic on the basis of the definitions of symbols, terms and formulas offered in work [3] let’s introduce definitions of:

Abstracts of type: 1) individual formula – type abstract, 2) if A is an individual formula, α – type abstract, then $\{\alpha A\}$ – type abstract, 3) there are no other type abstracts;

Determinants as the t def η form constructions, where t – individual term, η - type term;

Specifiers: objects, situations, events t spec A where t – indiviual term for objects, situations, events or type term, A – type abstract; actions ι: «$\beta 1$> <|$\beta 2$ », where t – individual term or type term, $\beta 1$ i $\beta 2$ – pre-condition and post-condition as situation specifiers; problems ι: «$\beta 1$> <|$\beta 2$ », where ι – individual term for TProblem object or type term $\beta 1$ and $\beta 2$ – pre-condition and post-condition as situation specifiers;

clause: expressions of the form A←K, where A is the sequence of atomic formulas of the language of the AP system, K is the only atomic formula.

The knowledge base of the AP system consists of:

1) sets of specifiers of the description of the initial situation and types;

2) sets of clauses that specify the properties of operations and predicates for individual objects and types;

3) a set of definitors and clauses that define the properties of types;

4) a set of clauses specifying the pre-conditions and post-conditions for action;

5) a set of clauses specifying the event component of the pre-conditions.

In the definition of the AP-system within the framework of the clause logic of the first order on the basis of the definitions of symbols, terms and formulas proposed in [3], let’s introduce the definition:

Programs of actions: 1) if ι – individual term of an action type, then ι - program of actions; 2) if y_1, y_2 - programs of actions then y_1, y_2 - program of actions; 3) if $\iota 1$ - individual term of an action type, $\varepsilon 1, \ldots \varepsilon n$ - situation type formulae $y1, \ldots yn$ - programs of actions, then case $(\iota 1, (\varepsilon 1, y1), \ldots, (\varepsilon n, yn))$ is a program of actions; 4) if $y1$ – program of actions, and $c1$ – situation type formula, then while $(y1, \varepsilon 1)$ - program of actions; 5) there are no other programs of actions;

Programs of actions specifiers: 1) if y_1 - schematic term, y_2 - program of actions, then y_1, y_2 - program of actions specifiers; 2) there are no other programs of actions specifiers;

clauses: expressions of A←K form, where A is the sequence of atomic formulas of the language of the PC system, K is the only atomic formula.

The knowledge base of the PC system consists of:

1) a set of specifiers for action programs;

2) a set of clauses that define problem-solving schemes;

3) a set of clauses that define the properties of the schemes.

Let’s consider the derivation procedure on the example of a natural case when queries take the form $<< \lambda 1 >, < \lambda 2 >>$ ’ where $\lambda 1, \lambda 2$ - abstracts of initial and target situations. The output procedure includes the following steps:

1) establishing an individual situation or type of situation in the AP system (if the request defines the individual terms of the initial and target situations);

2) installation of an action program in the PC-system (if the request defines the abstracts of the initial and target situations);

3) installation of the program of actions in PC-system (if in request types of initial and target situations are defined);

4) if necessary, to define the program of actions in the PC-system;

5) if necessary, to build an action program in the PC system;

6) if necessary, set the necessary type conversions.

To reduce the search, programs for popular queries are saved and known techniques are used, first of all, typification of statements, the method of analogies. To substantiate the correctness of the obtained derivation results, we can use the theorems on effective reachability proved in [3].

Methods of formal logic output system realisation

Construction of logic of transitions between states on the basis of logical components and lexical rules, should be set by the user in the convenient form. During the review of the available ready-made technical solutions that can provide display and implementation of the logic of transitions, the Camunda framework was chosen [5]. This framework includes the implementation of the transition mechanism, which as a separate service of our infrastructure with existing modifications to implement template and save behavior and input conditions provides solutions to technical problems provided by the tasks and greatly simplifies the complexity of creating such a class of systems. The efficiency of using this framework and performance results are given in [6].

Conclusion

Based on the analysis of messengers, chatbots, problems of their creation, use and development, an approach to automated creation of bots based on logical and linguistic models is proposed. Problems of formalization of processes of creation of bots are considered. One of the key features of this solution is the automation of the chain of actions, the implementation of which leads to the execution of the user's request.

The logical formalism as a basis of this decision is described, the formal definition of achievement of the purposes and conditions which the action plans providing achievement of the purposes should correspond is offered. A derivation procedure is proposed that allows to form action plans to solve problems formulated as pairs of initial and final states.

The implementation of the proposed solution will allow you to quickly create bots for a wide range of problems, formulated in terms of pairs of initial and final states, based on a combination of functions of the respective classes of objects, also given by pairs of initial and final states. Thus, you can quickly create bots to obtain the necessary information, place orders, provide access to educational resources, etc.

Further research should be continued in order to create and study bots to solve certain classes of problems, create technologies for effective natural communication of the bot with the user, create information technology to integrate the user's work with all messengers, social networks and information systems.

REFERENCES

1. Radziwill N. Evaluating Quality of Chatbots and Intelligent Conversational Agents [Електронний ресурс] / N. Radziwill, M. Benton. – 2017. – Режим доступу до ресурсу: https://arxiv.org/ftp/arxiv/papers/1704/1704.04579.pdf.

2. Most popular global mobile messenger apps [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://www.statista.com/statistics/258749/most-popular-globalmobile-messenger-apps/.

3. The approach to applications integration for World Data Center interdisciplinary scientific investigations / Grzegorz Nowakowski, Sergii Telenyk, Kostiantyn Yefremov, Volodymyr Khmeliuk // W: Proceedings of the 2019 Federated Conference on Computer Science and Information Systems, September 1–4, 2019, Leipzig, Germany [online] / eds. Maria Ganzha, Leszek Maciaszek, Marcin Paprzycki. – Warszawa : Polskie Towarzystwo Informatyczne, 2019. – (Annals of Computer Science and Information Systems, ISSN 2300-5963 ; 18). – S. 539-545. – doi: 10.15439/2019F71. – ISBN 978-83-952357-8-8 (Web).

4. Sergii Telenyk, Nowakowski Grzegorz, Eduard Zharikov, Vovk Jewhenii. Conceptual foundations of the use of formal models and methods for the rapid creation of web-applications // The 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. 18-21 September, 2019, Metz, France.

5. Camunda web site. Режим доступу до ресурсу: https://camunda.com/products/camunda-platform

6. The Activiti Performance Showdown Режим доступу до ресурсу: https://dzone.com/articles/activiti-performance-showdown