http://www.moderntechno.de/index.php/meit/article/view/meit42-01-015
DOI: 10.30890/2567-5273.2025-42-01-015

UDC 004.318

FEATURES OF DEVELOPING AN INTELLIGENT CHAT BOT ON THE TELEGRAM PLATFORM AND A MOBILE APPLICATION ON THE ANDROID PLATFORM

Burdaiev V.P.
c.ph.math.s., as.prof.
ORCID: 0000-0001-9848-9059
National Technical University "Kharkiv Polytechnic Institute",
Kharkiv, Kirpichova, 2, 61000

Abstract. *The purpose of this work was to develop an intelligent chatbot on the Telegram platform and an Android mobile application for online learning. This paper examined technological methods for creating chatbots and a mobile application using the example of choosing a technology stack for a web application. The first method involved implementing a KARKAS shell inference mechanism into the chatbot for creating knowledge base models. The second method utilized a programming interface that accepts requests from the chatbot, processes them, and sends them back to the chatbot using an API. The third method was based on the concept of a multi-layered knowledge base structure for a mobile application for online learning. The devices demonstrated high efficiency (90%) in preparing for exams in the subject "Cloud Computing and Its Applications."*

Key words: *online education, messenger, multilayer knowledge base, integration, business logic.*

Introduction.

Chatbots and mobile applications based on the knowledge base play an important role as online education services. They minimize human participation in online educational processes and provide an opportunity to study from anywhere in the world.

The object of the study was the online educational process, and the subject of the study was the development of intelligent tools: chat bots and a mobile application for the field of online education.

Modern development of the Internet considers distributed intelligent systems as qualitatively new technologies, the features of which are modeling of functional systems, use of dynamically developing ontology of the subject area, multi-agent choice of adaptive strategy of decision-making. One of the important directions in the field of intelligent information systems is the problem of providing online mode of user consultation with these systems. Various messengers demonstrate a high level of user attraction, compared with all other applications of other categories. The paradigm of integrating chatbots to work with intelligent systems is currently becoming

increasingly relevant. The current problem for chatbots is to create an inference engine that determines the relevance of knowledge for a given question.

Using a messenger as an interlocutor provides more opportunities for online consultation with an expert system via a smartphone. S. Russell et al. [1], C. Culbida et al. [2] consider one of the important areas in the field of intelligent information systems to be the task of providing online consultations to users of these systems. Messengers can be considered as a kind of browser, and their chatbots as web applications (with elements of artificial intelligence). Developing software based on artificial intelligence technologies is quite complex. Therefore, various prototyping tools are widely used to show clients how the chatbot will look and behave.

K. L. Ehsani et al. provided a taxonomy of chatbots based on the following features: goal-based chatbot; knowledge-based chatbot; service-based chatbot; answer-based chatbot [3].

A. Shahab et al. developed an intelligent chatbot that can understand and respond to user queries [4]. J. Walden et al. described the development of a chatbot designed to teach students in a web development course. The chatbot was shown to be effective in providing accurate answers related to secure programming through inductive reasoning [5].

O. Ivashchenko et al. developed a chatbot to provide quick and intuitive access to information about academic procedures, communication channels, scholarships, documents, and other common issues related to students' interactions with the department and its website. The core of the system is a structured multi-level intent architecture, in which each group of intents corresponds to a topic category, such as admission, documents, or course schedules. This allows the bot to maintain the context of the conversation, provide accurate request routing, and reduce ambiguity in user interactions [6].

M. Bagchi discussed an integrated modular machine learning framework Rasa Stack. A typical Rasa Stack workflow in a chatbot includes: a connector module, a dialog management module with an API call, an output module to convey the chatbot response [7].

M. N. Hamidah et al. developed Android-based mobile applications. In their works, calculations in applications are performed using the forward chaining method [8].

The aim of this work was to develop an intelligent chatbot on the Telegram platform and a mobile application on the Android platform in the online educational industry.

The first problem is the integration of the chatbot with the KARKAS (Knowledge + Acquisition + Relevance + Knowledge + Accumulation + Shell) modules [9].

The second problem is to develop an API in the PHP programming language and an intelligent chat bot based on a finite state machine.

The third problem is to develop an asynchronous mobile application in the field of online education based on a multi-layered knowledge base structure.

Main text.

The first problem was solved by integrating the KARKAS shell inference engine into the @es_info_tech_karkas_bot chatbot code on the Telegram platform. As a result of the integration, the chatbot inherited the shell's hierarchical functional system and knowledge base filtering. The integration of the chatbot with the shell's consulting agent involves exchanging messages between them, that is, sending and receiving requests to work with Telegram servers. Chatbot modules exchange messages with the shell to perform the following operations: activating buttons, checkboxes, and radio buttons, as well as sending and receiving messages between visual objects on the shell's agent-consultant form. The iterative consultation process with the chatbot continues until the shell's inference agent receives a result.

The integrated chatbot with shell inference machine operates using the following mathematical model $< G,$ F, R, S $>$, where G $= \{G_{t0}, G_{t1}, \dots, G_{tn}\}$ is the set of chat bot goals, $F = \{F_{t0}, F_{t1}, \dots, F_{tn}\}$ is the set of facts characterizing the state of the chat bot, R $= \{R_{t0}, R_{t1}, \dots, R_{tn}\}$ is the set of rules for controlling the chat bot and S $= \{S_{t0}, S_{t1}, \dots, S_{tn}\}$ is the set of chat bot states. Then S_{ti} is the state of the goals, the fact base (data) and the rules at the time t_i, in other words, $S_{ti} = \{G_{ti}, F_{ti}, R_{ti}\}$ is the set of all goals G_{ti}, facts F_{ti} and rules R_{ti} that determine the behavior of the

chatbot at a fixed time t_i.

Then the trajectory of the chatbot's functioning can be considered as a change in its states at points in time. The set of all possible states of the chatbot forms its subject area space. Facts F_{ti} are understood as the designated properties of the chat bot, which are inherited from the rules.

Each rule from the set R corresponds to the state S_{ti} of the chat bot to which the given rule is applicable. As a result of the execution of the rule R_{ti}, the values of F_{ti} change, that is, the fact F_{tj} and the state of the chat bot $S_{tj} = \{G_{tj}, F_{tj}, R_{tj}\}$. are formed. Thus, each rule is a mapping of one state to another.

Depending on the state S_{tj}, at the moment $t_j \in [t_{beg}, t_{end}]$ the inference engine φ_{ti} selects the rule that is necessary to solve the problem. In other words, to control the chat bot, the mapping φ_t for some fixed t of the chat bot state space onto itself is used: $\varphi_t : S \to S$.

The task of the chat bot operation is to transfer the chat bot from the initial state S_{t0} using the mapping φ_t to some new state, defined as the target state S_{tg}. For the process of functioning of the chatbot, a display of the shift per unit of time is defined. This action is like a set of samples of the chatbot states through successive discrete time intervals of duration t. The target state of the object S_{tg} is defined by the expression: $S_{tg} = \varphi_{tj}\left(\varphi_{ti}\left(\varphi_{tk}\left(\dots\left(\varphi_{t0}\left(S_{t0}\right)\right)\right)\right)\right)$. The trajectory $\langle \varphi_{tj}, \varphi_{ti}, \varphi_{tk}, \dots, \varphi_{t0} \rangle$ of the iteration of the mapping φ_t is an algorithm for solving the problem for an intelligent chatbot.

Integrating the shell output mechanism with the chatbot allows it to inherit the shell's user interface. In this case, the sequence of questions posed to the user is generated dynamically according to the rules of the knowledge base. Because the @es_info_tech_karkas_bot chatbot is integrated with the shell, it is monolithic.

The user interaction scenario consists of the following actions: launching the es_info_tech_karkas_bot.exe application on a local computer or a virtual machine in the cloud; selecting the @es_info_tech_karkas_bot chatbot in the Telegram messenger

and entering the command /help or /start; the chatbot launches the shell consulting agent, which activates the shell interface and creates a hierarchical functional system for interacting with the user. The shell dialogue agent then sends a message with the question text and answers. The chatbot receives the message as a JSON object, parses it, displays the message in the chat, and waits for a response from the user. The user selects or enters a response in the chat. The chatbot sends it to the shell output mechanism. The shell consultation agent receives the message and passes it to the interface mechanism. The shell dialogue agent performs the next step in accordance with the constructed hierarchical functional system for dynamic consultation [10]. Note that the consultation goal is formed during the dialogue with the user. In other words, the choice of goal depends on the user's responses to the chatbot's questions. The iterative consultation process continues until the shell interface mechanism receives a result. The user can interrupt the consultation at any time using the /quit command.

Since the shell is a monolithic application, the integrated chatbot is also a monolithic application. Therefore, scaling such a chatbot is quite a complex task.

The second approach to creating an intelligent chatbot was based on the creation of microservices in the PHP programming language. An API is a scalable programming interface that allows two applications to interact with each other. The first application is hosted on a server, and the second is hosted on a platform, for example, Telegram. Cloud technologies use APIs to connect loosely coupled microservices. For example, an API tells the application what data and messages the microservice needs and what results the microservice can provide.

The architecture of the PHP application programming interface for a chatbot on the Telegram platform is based on the use of webhook support. Using webhooks allows you to avoid embedding data structures and methods for processing them in the chatbot code. Thus, the chatbot business model is based on the exchange of information in the form of messages between the chatbot and the API.

The architecture of an intelligent chatbot is a complex system that must not only be scalable and fault-tolerant, but also effectively manage user requests. To receive and transmit messages between the chatbot and the backend, the JSON format and

HTTP methods are used: get, post, put, delete.

The database (MySQL) is used to store chatbot messages and consists of the following tables: users (contains information about users), test_aws (contains information for the test of knowledge of Amazon technology), test_azure (contains information for the test of knowledge of MS Azure technology), exam (contains information for the exam of knowledge of Amazon and MS Azure technologies).

The knowledge base (MySQL) is used to manage the chatbot's business logic and consists of the following tables: test_ws (contains information for testing knowledge on choosing a web stack), knowledge_ws (contains rules for choosing a technology stack for a web application). When receiving a Telegram account, the chatbot's scalability is provided by the API and the platform itself. JSON Web Tokens are used for data encryption, authentication, and authorization.

The Index.php file is the entry point to the intelligent chat bot. The file structure of the chat bot API project is as follows: the CONFIG directory contains the config.php file, which includes the knowledge base credentials for interacting with MySQL and a number of getTelegramRoutes(), getRoutes() functions for managing the chat bot; the LOADER directory contains the API boot files, which contain the ClassLoader, Route and other classes; the DATAKNOWLEDGE directory contains files that interact with the knowledge base; the MODEL directory contains the files: users.php, webstack.php, aws.php, model.php, expertsystem.php, exam.php, in which the business logic is moved to inherited classes; the CONTROLLER directory contains the main files: the CommanderController.php file, which contains all the necessary methods for interacting with the chat bot through various menu items; the TelegramController.php file contains general utility methods for interacting with the Telegram platform.

The API-based intelligent chatbot model is implemented as a deterministic finite state machine. The initial state S_0 of the chatbot is defined by the following set of states: S_0 = {Testing (for web stack), Testing (for AWS), Testing (for MS Azure), Exam: cloud computing AWS + MS Azure, More about testing, Mobile testing (for MS Azure), Expert system (for web stack)}. To activate the initial state of the chatbot, use the command: /start. To call the final state of the chatbot, use the command: /quit.

The {Testing (for web stack)} state is designed to test the user's knowledge of how to choose a technological web stack for a web application. The test contains 20 questions. When the user selects the {Testing (for web stack)} state, the chatbot moves to the following states: {Number question (1, 2, ..., 19, 20)}, {Receiving response from user}, {Evaluation}, {To the questions}. The {Number question (1, 2, ..., 19, 20)} state allows the user to select a test question in any order and specify multiple answers to the question. The answers are specified as numbers. When a question is selected, the chatbot sends an SQL query to the backend database and displays the query result as a question with answers. The user enters an answer and the chatbot moves to the {Receiving response from user} state. The chatbot then sends the user's response to the database and moves to the {To the questions} event. If the user has already answered the question, the chatbot informs him about it and moves to the {To the questions} state.

The {Evaluation} state allows the user to view the answers to the questions and the test score on a 100-point scale. The {Home} state returns the chatbot to the initial state S_0.

The activation of states and transitions between them are carried out in a similar manner for the {Testing (for AWS)} state. Here, 30 questions are used to test the user's knowledge of AWS cloud technologies. For the {Testing (for MS Azure)} state, testing is carried out using 49 questions.

The {Exam: cloud computing AWS + MS Azure} state condition is designed to obtain an exam score on a 100-point scale for knowledge in both AWS cloud technologies and MS Azure. The functionality of the condition is similar to the functionality for testing, but there is a time limit for the exam, which is 1200 seconds for 40 questions.

The {More about testing} state allows you to go to the {Go to https://www.en.it-karkas.com.ua} state, which has access to the main page of the KARKAS shell site. The chat bot state {Mobile testing (for MS Azure} allows you to go to the state {Go to expert systems, testing for IT}, which has access to the Google play web page for the TECHSTACK mobile expert system.

The state {Expert system (for web stack)} is intended for online consultation of chat bot users on the choice of a technological web stack for a web application based on the expert's knowledge. The following states are used for this: {Web stack (11)}, {Result}, {Home}. The state {Web stack (11)} allows you to ask the user 11 questions to determine the facts on the choice of a technological stack for a web application. The transition between states is the same as when using testing states. The state {Result} allows the user to analyze the following at any time during the consultation: how the fact was formed and what knowledge base rules were used to obtain the result of the consultation. The state {Home} returns the user either to continue the online consultation or offers to go to the initial state of the chat bot.

The advantages of using a deterministic finite state machine are that the logic of the dialogue with the user is simplified and the functionality of the chatbot is expanded. The functionality of the chatbot is divided into the following microservices: user registration service, database services and business logic service for managing the chatbot (Figure. 1).

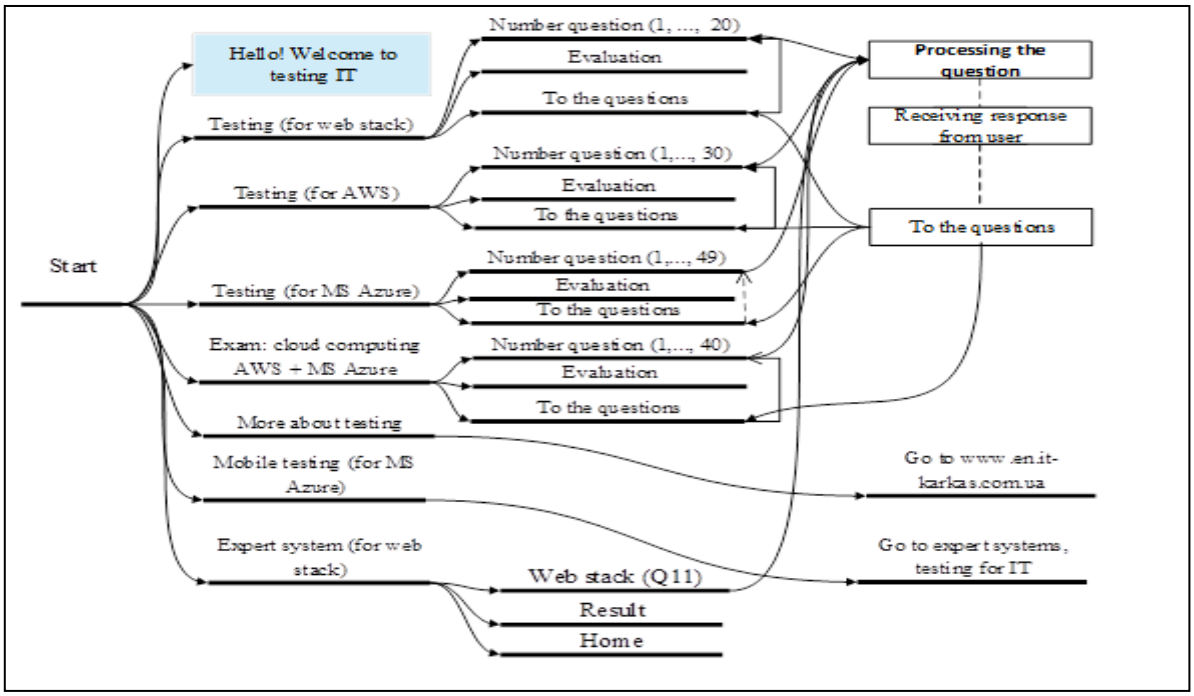

Figure 1 - Diagram of a chat bot @itbvp_bot

Authoring

For the API-based intelligent chatbot @itbvp_bot, the question sequence strictly corresponds to the business logic of the chatbot's state machine.

The user interaction flow with the chatbot using the API is as follows: the user sends a request (e.g., /start or clicking the chatbot button) to the API; the CommanderController receives a message; in accordance with the logic of the getTelegramRoutes function from the config.php file, a method is called to process the knowledge base business logic; the CommanderController sends a message to the chatbot for further user action.

For the third approach to online education, a mobile intelligent application TECHSTACK (Expert system for IT) was developed based on the Android platform. This app was used by students in a hands-on setting to select a technology stack for a web application. The app is available on Google Play for consulting users: project managers, developers, or decision makers in the field of corporate technology. The knowledge base structure is represented by the following classes: tech stack, Web application type, cost of developing a Web application, Web application sizes, Web application performance [9].

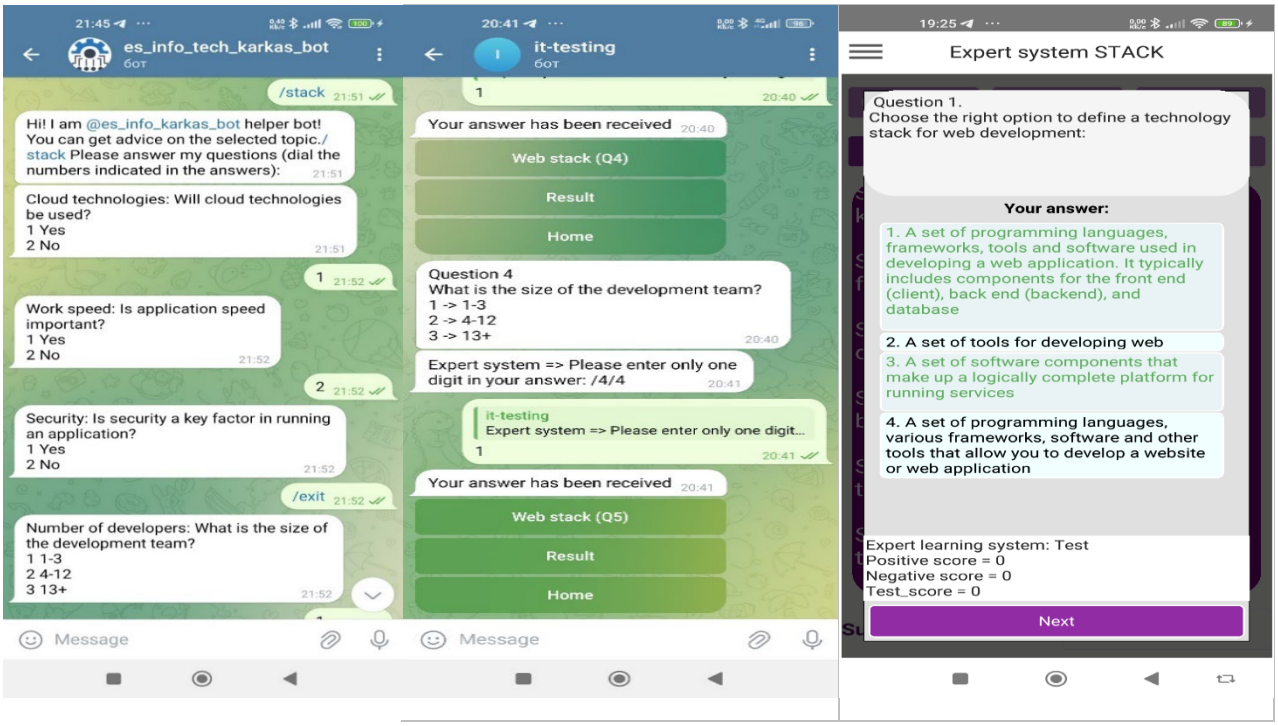

Figure 2 - Screenshots of interactive interfaces developed for online education

Authoring

The following screenshots show fragments of the dialog interface: on the left in the figure is the dialog interface of the integrated chatbot @es_info_tech_karkas_bot with a shell KARKAS; in the center of the figure is the dialog interface of the chatbot @itbvp_bot; on the right in the figure is the dialog interface of the intelligent application TECHSTACK (Figure 2).

Summary and conclusions. Have been considered examines the development of specialized online learning tools that have demonstrated effective motivation for students to acquire competencies in internet technologies.

Were received:

1. The integrated chatbot @es_info_tech_karkas_bot with the KARKAS shell inherits the hierarchical functional system of knowledge base representation and is a monolithic application.

2. The intelligent API-based chatbot @itbvp_bot belongs to the class of distributed expert systems.

3. The TECHSTACK mobile application allows students to conduct individual testing and prepare for exams using an active multi-level knowledge base.

The chatbot @itbvp_bot and the TECHSTACK mobile application can be used as standalone tools for teaching cloud technologies, accessible online 24/7.

The chatbot and mobile application enabled individual testing and active student preparation for exams. Third-year (33) and fourth-year (21) students participated in the experiment. The following courses were selected: "Continuous Integration and Deployment (CI/CD) Technologies" and "Cloud Computing and Its Applications." The first-year exam results were distributed as follows: 2 students scored 95 points, 12 students scored 90 points, 10 students scored 80 points, and 9 students scored 75 points. The second-year exam results were distributed as follows: 2 students scored 95 points, 10 students scored 90 points, 3 students scored 85 points, and 6 students scored 75 points. Many students (20%) used only the chatbot, as the mobile app is developed only for the Android platform.

A further goal of the study is to develop an intelligent chatbot on the Facebook Messenger platform using the API.

References:

1. Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach, Peter.Norving.

2. Culbida, C. I., Mylonas, P., Troussas, C., Krouska, A., & Sgouropoulou, C. (2025). Knowledge Management Systems: A Review of Artificial Intelligence Integration and Technologies. In: Artificial Intelligence Applications and Innovations. AIAI 2025 IFIP WG 12.5 International Workshops. AIAI. IFIP Advances in Information and Communication Technology, 754 (pp. 105–117). Springer, Cham. DOI: 10.1007/978-3-031-97313-0_9

3. Ehsani, K. L., Rhythm, E. R., Mehedi, M. H. K., & Rasel, A. A. (2023). A Comparative analysis of customer service chatbots: efficiency, usability and application. Computer Applications & Technological Solutions. DOI: 10.1109/CATS58046.2023.10424303

4. Shahab, A., Singh, S., Hazela, B., & Singh, V. (2024). Intelligent Chat Bot. Journal of Management and Service Science, 4(65), 1-5.

5. Walden, J., Caporusso, N., & Atnafu. L. (2022). A chatbot for teaching secure programming. A chatbot for teaching secure programming. In proceedings of the EDSIG Conference. (pp.1-10). https://iscap.us/proceedings/2022/.

6. Ivashchenko, B. V., S. Filip, S.& Ratushnyi, B. V. (2025). Development of an educational chatbot with a contextual intent system on the dialogflow platform. Bulletin of National Technical University KhPI Series System Analysis Control and Information Technologies, 1(13), 17-24. DOI: 10.20998/2079-0023.2025.01.03

7. Bagchi, M. (2020). Conceptualising a library chatbot using open source conversational artificial intelligence, Journal of Library & Information Technology, 40(6), 329-333.

8. Hamidah, M. N., Arizal, A., Lukas, A., Zainal, R. F., & Rahajoe, A. D. (2021). Decision support system for inheritance distribution according to islamic law using the forward chaining method. Journal of Electrical Engineering and Computer Sciences, 6(10), 1125-1134.

9. Burdaev, V. (2024). Features of Intelligent Systems Development for Platforms Telegram and Android. In proceedings of the ITEST 2024, 2 pp. 156–171, DOI: 10.1007/978-3-031-71804-1_11

10. Burdaev, V. (2021). On one approach to building a temporal model of the knowledge base. In proceedings of the 5th international conference on computational linguistics and intelligent systems, (pp. 1039–1048). Lviv: Computational Linguistics and Intelligent Systems.

Article sent: 09.12.2025