UDC 004.89

AGENT-DRIVEN APPROACH TO ENHANCING E-LEARNING EFFICIENCY

Dr.Sci. N. Axak ORCID: 0000-0001-8372-8432
Kharkiv National University of Radio Electronics, Ukraine
E-mail: nataliia.axak@nure.ua
Ph.D. M. Kushnaryov ORCID: 0000-0002-3772-3195
Kharkiv National University of Radio Electronics, Ukraine
E-mail: maksym.kushnarov@nure.ua
A. Tatarnykov ORCID: 0000-0002-1632-8188
Kharkiv National University of Radio Electronics, Ukraine
E-mail: andrii.tatarnykov@nure.ua

Abstract. *The paper discusses the problems of modern distance learning, such as individualization of the learning process. The need to improve the methods of supporting remote learning to enhance the quality of education is substantiated. The study aims to improve the quality of education and the organization of the educational process in distance learning. A set of interrelated tasks is proposed, the solution of which is to organize effective interaction between the teacher and the learning management system (LMS) using an agent-based approach. The main result that defines the novelty of the work is the formalization and integration of the following processes: (i) generating real-time suggestions to the teacher to control the performance of tasks during exams or electronic testing; (ii) monitoring students' learning during the semester with the possibility of changing the learning trajectory; (iii) monitoring parents' presence in online classes; (iv) generating recommendations for management and other stakeholders to improve online learning. Also presented are verification of the formal specification of an agent-based support system decision-making for distance learning using the SPIN tool and and the Promela modeling language. Verification is carried out using the formulas of linear temporal logic (LTL), which allows you to check the correctness of the interaction between agents and the fulfillment of key system properties, such as timely data processing, response to violations of students' compliance with testing rules and teachers' recommendations. The modelling of the prototype of the proposed system confirms the effectiveness of its use as a means of studying the organization of the educational process. It shows how agents can help collect and analyze data on how effectively electronic resources are used for learning by students and teachers.*

Keywords: *Learning management system, monitoring, learning process, agents, cloud computing, distributed system temporal logic specifications, alternating-time temporal logic, verification*

1. An overview of the accomplishments of multi-agent systems in education

Decision-making is a crucial step in the educational process, as it affects the quality of learning and student success. Some issues can arise when making decisions in the educational process: Insufficient processing or the absence of necessary data can lead to the decision-making process being insufficiently informed and reducing its effectiveness; limited resources, such as budget, staff, and time, can force decisions that are not the best for the quality of education and student success; the lack of education staff's qualifications can result in misunderstanding and data analysis, which can result in incorrect decisions; the variety of students in a classroom can complicate decision-making, as their needs and interests may differ; students who lack motivation can decrease the effectiveness of learning and result in lower-quality solutions; the quality of education and student success can be negatively impacted by social issues, such as poverty, violence, and discrimination, which can complicate decision-making. These problems can reduce the efficiency of decision-making in the educational process. The primary strengths of utilizing agents for decision making lie in their ability to: process automation - agents can help automate complex decision-making processes, which can increase productivity and reduce time spent on decisions; data collection and analysis - agents can help collect and analyze large amounts of data, which allows you to identify trends and patterns that can be used to make better decisions; intelligent decisions - agents can be trained to make decisions based on previous experiences and training using machine learning and other technologies, allowing them to make better decisions based on the analysis of many factors; individualized approach - agents can help to develop customized solutions for participants in the queuing process, which allows for a more personalized approach and meet the needs of different groups.

Consequently, applying agents in the educational process is both relevant and capable of enhancing decision-making quality.

2. Related works

Recently, research related to agent-based decision-making in education has taken a big step forward. The book [1] focuses on Intelligent Tutoring Systems, which are one of the most common uses of agents in education for decision-making. The discussion chapters in this book look at topics through the lens of the Generalized Intelligent Framework for Tutoring (GIFT). Tutoring techniques, strategies, and tactics play a central role in the development of GIFT. The techniques within GIFT are expected to be implemented as software agents, where the agent monitors the learner's progress and the learning context to determine whether best practices (agent policies) have been followed or violated. Over time, the agent will learn to apply the agent policies in a way that optimizes learning and performance.

Many researchers interested in education and learning around the world have realized the potential of data analytics and artificial intelligence and have intensified research around them. This interest has led to the emergence of the fast-growing and multidisciplinary field of learning analytics and increased research into how artificial intelligence can support learning and teaching methods [2]. Artificial intelligence in education (AIEd) opens up new opportunities, potential, and challenges in education. Throughout its history, AIEd has undergone several paradigm shifts, which are characterized in [3] by three paradigms: AI-driven, learner as recipient, AI-supported, learner as collaborator and AI-empowered, and learner as leader. In the three paradigms, artificial intelligence methods are used to solve education and learning problems in different ways. One of the potential roles of artificial intelligence in education is to provide opportunities to augment human intelligence, with AI supporting us in decision-making processes rather than replacing us through automation. The authors of the study [4] present a case study in the context of debate tutoring, in which they use prediction and classification models to increase the transparency of intuitive decision-making processes by expert tutors for extended reflection and feedback. The trend of AIEd is evolving to empower learners and personalize their learning, enable learners to reflect on their learning and inform artificial intelligence systems to adapt accordingly, and lead to the iterative development of learner-centered, data-driven personalized learning [5]. Study [6] proposes agent-based virtual and intelligent recommendations that require information about users' profiles and preferences to recommend the right content. They applied natural language processing (NLP) techniques and semantic analysis approaches to recommend course selection for e-learners and teachers. The content-based recommendation method provides content suggestions related to students' requests and preferences. The use of social media from an educational perspective makes it possible to provide a user-friendly interface for recommending the highest level of interaction in terms of collaboration between users and contacts. E-learning is the threshold of social network-based learning (SN-Learning). SN-Learning consists of a new term introduced in [7] and includes e-learning systems with social networking characteristics or learning through social networking platforms. The authors provide guidelines for designing and implementing SN-Learning platforms using artificial intelligence and modeling techniques. Article [8] explores the possibility of disaggregating query information in online lectures of an e-learning system or course recommendations. Information organization includes reading, parsing, and classifying question messages. Data extraction is a type of surface content processing. It finds a set of predefined applicable content in the feature language archives and performs general language processing using artificial intelligence strategies. Learners often have difficulty finding and retrieving relevant materials to support their learning goals because they lack the subject matter knowledge to create effective queries that convey what they want to learn. In addition, the unfamiliar vocabulary often used by subject matter experts makes it difficult to match a learner's query with relevant learning material. The authors of [9] solve these problems by introducing an innovative method that automatically creates basic knowledge for a learning domain. The effectiveness of the proposed method is evaluated using a collection of machine learning and data analysis documents. The article [10] identifies the features of human and artificial intelligence decision-making along five key contingency factors: specificity of the decision space, ability to interpret the decision-making process and results, size of the alternative set, speed of decision-making, and repeatability. Based on a comparison of human and artificial intelligence decision-making along these dimensions, the article creates a new framework that describes how both decision-making methods can be combined to optimally improve the quality of organizational decision-making. The authors of [11] propose future research directions in a triple perspective: key methodologies for Large Scale Decision Making (LSDM), AI, and data fusion for LSDM. In [12], the authors describe a meta-reasoning policy that can be implemented by a team of agents to make effective control decisions at the meta-level based on the availability of communication in the environment. The authors synthesize the meta-reasoning policy as a solution to the reactive synthesis problem involving the level of communication in the environment and the choice of the agent's algorithm. The authors of [13] argue that in a multi-agent environment, it is appropriate to ask what behavior the system will exhibit under the assumption that agents act rationally, following their preferences. They promote a paradigm of rational verification for multi-agent systems, as an analog of classical verification. The authors tried to automatically determine whether the given properties of a system, expressed in the form of temporal logic formulas, will be preserved in this system under the assumption that the system components (agent) behave rationally by choosing (for example) strategies that form a game-theoretic equilibrium. The article [14] aims to provide a comprehensive view of the relationship between agents and multi-agent systems (MAS) on the one hand, and logic-based technologies on the other, by making them the subject of a systematic literature review. The resulting technologies are discussed and evaluated from two different perspectives: MAS and logic-based. The paper lists the most common logic-based technologies (47 in total) for MAS, but only a relatively small number of them conform to major technology standards. Temporal logics have been widely used in model checking as a formalism for reasoning about the execution of computer systems. They are powerful enough to define most of the properties that can be verified by reactive systems, while also providing very efficient verification algorithms [15]. Temporal logic and model checking have had a major impact on computer science and have been applied in many industrial cases. Several attempts have been made to extend temporal logic to multi-agent systems where multiple components interact, for example, Computation-Tree Logic (CTL) can only express the existence (or not) of an execution of a global system with certain properties, where the goal is to quantify the possible behavior of individual components interacting in the system. In 1997, CTL was extended to Alternating-time Temporal Logic (ATL) with the introduction of strategy quantifiers. In ATL, strategy quantifiers express the existence (or not) of a behavior of one of the agents (or a coalition), so that any final execution in the global system satisfies this property. Study [16] is related to multi-agent logic and its application in computer science. The authors work with multi-agent logic based on relational models. They determine that time availability relations can have gaps or places of forgotten time. The authors of [17] study the problem of learning to satisfy temporal logic specifications with a group of agents in an unknown environment that can exhibit probabilistic behavior. From a learning perspective, these specifications provide a rich formal language with which to capture tasks or goals, while from a logic and automated verification perspective, the implementation of learning capabilities allows for practical applications in large, stochastic, unknown environments. The temporal logic of actions (TLA) — is a logic for specifying and reasoning about parallel systems. The systems and their properties are represented in the same logic, so the statement that a system conforms to its specification and the statement that one system implements another are expressed by logical consequence. TLA is very simple; its syntax and full formal semantics are summarized in about a page. Report [18] introduces TLA and describes how it is used to define and verify parallel algorithms. Education systems include a variety of components such as learning management, progress tracking systems, e-textbooks, etc. The challenge is to interact and integrate intelligent agents with these systems to ensure that they work together effectively. For a system to be successful, agents need to be able to quickly adapt to new requirements and change their behavior accordingly. Thus, the fundamental role of logic-based technologies in MAS today is to meet the need for intelligence that characterizes agent abstractions, i.e., the cognitive abilities required by components of a distributed intelligent system. However, logic-based technologies can rarely be considered mature enough to meet the requirements of industrial and real-world domains. Research on the use of intelligent agents in decision support systems in the educational process reveals new opportunities and approaches that help improve learning and provide an individual approach to students. But despite the significant progress in the use of an agent-based approach to support decision-making in the educational process, there are some problems that remain to be fully resolved. Although intelligent agents show the potential to improve the educational process, their widespread adoption and accessibility to all educational institutions can be a challenge. This is due to high development and implementation costs, the need for specialized skills and infrastructure. Education systems include a variety of components such as learning management, progress tracking systems, e-textbooks, etc. The challenge is to interact and integrate intelligent agents with these systems to ensure that they work together effectively. Some intelligent agents may be limited in their flexibility and adaptability to the changing needs of students and teachers. For a system to be successful, agents need to be able to quickly adapt to new requirements and change their behavior accordingly. The use of intelligent agents in the education system raises ethical and privacy issues. The collection and processing of students' personal data may raise privacy concerns and the use of this data without authorization or in countries other than its intended destination. It is also important to ensure active interaction and involvement of students in the process of using intelligent agents. Uncontrolled use of agents can lead to student passivity and loss of motivation to learn. Given these problems, further research and development of intelligent agents in the educational process is aimed at solving these issues, ensuring accessibility, flexibility, ethical use and involvement of all participants in the educational process. The objective of this work is to elevate the standard of decision-making in distance learning by examining video data on student behavior.

To achieve this aim, the following tasks are completed:

• design a model of an agent-based decision support system for distance learning, including its elements and their interaction;

• determine user requirements using collected data and emerging trends in education;

• formalize the recommendation process for supporting teacher decision-making;

• verify that the system works correctly in all possible scenarios, including detecting violations, blocking access to tests in case of using prohibited tools, sending warnings, and processing appeals correctly.

3. The design of an agent-driven decision support system for online learning

This work introduces a system that combines multiple components: a video surveillance subsystem designed to detect actions or circumstances that may suggest potential violations during e-testing [19]; a multi-agent system where agents engage with the video surveillance system in real time [20], analyze gathered data, and offer recommendations; temporal logic for defining the system's real-time responses to different events and states; a decision-making system that autonomously reacts to identified violations; and automated mechanisms to address violations, such as sending alerts to administrators or instructors. We propose a model for an agent-oriented decision support system (AoDSS) for distance learning, offering a conceptual view of the system's architecture and its core principles. This model leverages the agent-based paradigm to enhance and optimize remote learning, with agents serving as key contributors to the organization and refinement of the learning environment. The proposed model of a decision support system for distance learning is based on set theory and temporal logic:

$$AoDSS = <S, A, T, L, R>, \tag{1}$$

where $S=\{s_1,s_2,\dots,s_M\}$ – a set of students, where each student $s_i \in S$ $(i=\overline{1,M})$, $A=\{a_1,a_2,\dots,a_N\}$ – is a set of agents, where each agent $a_j \in A$ performs certain functions in the system, such as providing recommendations, monitoring progress, analyzing results, etc. $(j=\overline{1,N})$; T – a set of learning materials or tasks, where each element $t_k \in T$ presents a separate learning task or material $(k=\overline{1,K})$; L – a set of knowledge or competence levels, де $l_u \in L$ determines the level of success of each student $(u=\overline{1,U})$; R – is a set of recommendations or actions proposed by the system, where $r_\varphi \in R$ meets specific agent recommendations for students $(\varphi=\overline{1,\Psi})$

Key elements of the model include. Defining the roles and functions of agents within the system, such as conducting requirements analysis, managing the training process, monitoring progress, facilitating interactions, and more. Mechanisms for agent interaction and information exchange to support decision-making. Adaptation mechanisms that enable agents to tailor the learning experience and offer recommendations based on the unique characteristics of each student. Identification of the technologies and platforms utilized for the implementation and coordination of agents within the system. Mechanisms for evaluating and tracking student progress through the use of agents and their functionalities. The agent-based decision support system model for distance learning is designed to optimize the educational process, enhance communication, and assist students in achieving superior outcomes. The goal of the system is to ensure that students' results improve through optimization of the learning process and personalized recommendations.

$$G(F_L(s_i)\uparrow), \quad (2)$$

that is, at each moment of time, the level of students' knowledge increases or remains the same, which means the effectiveness of agents in the system. Here G —temporal operator "Globally" (or "Always"). It means that a certain condition must be fulfilled at every moment of time in the future, i.e. what is happening now must continue at all subsequent moments of time; $F_L(s_i)$ — is a function that maps a student sss to his or her current level of knowledge or competencies. In other words, $F_L(s_i)$ means the level of knowledge of the student s_i at a certain point in time; ↑ means "increase" or "improvement", in this context, this symbol means that the student's knowledge level is increasing. Thus, expression (2) means that at all times (at any given time), the student's knowledge level si is either increasing or at least not decreasing. In other words, the system must ensure that the student's knowledge is constantly improving or remains at the same level, without deterioration. Expression (2) formalizes the requirement for the system: to ensure continuous improvement or stability of students' knowledge at all stages of learning. Each action or function has its own mapping between sets, and the dynamics of the system is described by temporal logic to determine the interaction between agents and students and optimize the learning process. Functions and mappings between sets in this model describe the relationships between elements of different sets (students, agents, learning materials, etc.) and reflect how each element of one set interacts or is related to elements of another set. This allows you to formalize the processes taking place in an agent-based decision support system for distance learning. Function. $F_A: S \to A$ відображає кожного студента $s_i \in S$ на певного агента $a \in A$, який відповідає за його навчальний процес. Іншими словами, кожен студент отримує підтримку від конкретного агента, який виконує функції, пов'язані з аналізом вимог студента, організацією його навчання, моніторингом його прогресу і наданням рекомендацій. Якщо студенту s_i призначається агент a_j, який буде моніторити його успішність і надавати необхідну підтримку, тому $F_A(s_i) = a_j$.

Function. $F_T: S \to T$ displays each student $s_i \in S$ for certain training materials or assignments $t_k \in T$, that he or she has to perform or learn. These materials can be selected according to the student's level of knowledge or needs. That is, the student s_i a task is assigned t_k, which corresponds to his level of training: $F_T(s_i) = t_k$.

Function. $F_L: S \to L$ connects every student $s_i \in S$ with his level of knowledge $l_u \in L$, which reflects his or her success or progress in learning. The level of knowledge changes depending on the learning outcomes and the effectiveness of the tasks. which reflects his or her success or progress in learning. The level of knowledge changes depending on the learning outcomes and the effectiveness of the tasks s_i has a level of knowledge l_u, то $F_L(s_i) = l_u$. As a student improves their performance, this feature can be updated, for example, $F_L(s_i) = l_{u+1}$. Function. $F_R: S \to R$ maps students to specific recommendations provided by agents to improve their learning. The recommendations can be individualized and take into account the student's progress, needs, and characteristics. The agent provides the student with s_i recommendation r_φ, which involves reviewing certain training material or performing additional exercises $F_R(s_i) = r_\varphi$.

These functions describe how the system manages learning processes, assignment selection, progress monitoring, and recommendation. They formalize the relationship between students and the system, allowing the system to provide personalized learning paths and improve the learning experience based on student data. Through these functions, the model determines how each student will receive personalized support to improve their performance in distance learning.

3.1 Multi-agent system

In formula (3), a multi-agent system is represented as a set of three components:

$$MAS = \{A, Act, E\}, \tag{3}$$

which consists

- a set of program agents A = {A_S, A_L, A_D, A_M, A_Q}, that operate in the environment Env, where A_S – student agent monitors the activity and academic progress of students, collects data on completed assignments, tests, and other information for further analysis; A_L – the teaching agent monitors courses, assignments, and student feedback, generates recommendations for improving materials and teaching methods to enhance teaching and decision-making in the educational environment; A_D – the data analytics agent is responsible for processing and analyzing data collected from students and teachers, identifying patterns, trends, and important connections to support decision-making; A_M – the system agent manages the interaction between other agents, distributes tasks, ensures the continuity of the system and coordination of processes; A_Q – the survey agent sends questionnaires to users, collects responses and stores them in the database;
- a set of actions of agents $Act(A_i)=\{a^i_1, a^i_2, \ldots, a^i_n\}$; i=S,L,D,M,Q; for the student A_S:$Act(A_S)$={complete_task, pass_test, review_material}; for the learning agent A_L:$Act(A_L)$={generate_recommendation, analyze_reviews}, for the data analysis agent A_D:$Act(A_D)$={ process_data, detect_patterns}; for the system agent A_M:$Act(A_M)$={ distribute_tasks, ensure_synchronization}; for the survey agent A_Q:$Act(A_Q)$={ send_survey, collect_responses}; actions for each agent are defined as: $Act(A_i) \subseteq Act, \forall A_i \in A$;
- a set of possible states of the environment, which changes depending on the actions of agents and events occurring in the system E= {E_{Mn}, E_{Md}, E_{Un}} – a set of

states of the environment of Student behavior monitoring subsystem (E_{Mn}), the environment of LMS Moodle (E_{Md}) and the environment of University Web portal (E_{Un}), ($E_j = \{e_1^j, e_2^j, \dots, e_m^j\}, j = \overline{(Mn, Md, Un)}$). For example, the set of states E_Md may include the following elements: e_1 – all tasks are completed, e_2 – the student needs help, e_3 – all reviews are collected, e_4 – the analytical report is ready.

The transition between states of the environment depends on the actions of agents. Let's introduce the function of transition between states:

$$\delta : E \times Act \to E.$$

This function determines how the agent's action affects the state of the environment. For example:

$$\delta(e_i, a_j) = e_{i+1}.$$

This means that after performing an action a_j (for example, completion of a task), the state of the environment changes from e_i (all tasks completed) on e_{i+1} (student needs help).

Recording changes in environment states over time $G(\delta(e_i\ a_j) = e_{i+1})$ means that always after performing an action a_j, the state will change from e_i on e_{i+1}.

For example, an agent A_S performs an action a_1 (completes the task), which changes the state of the environment from e_1 on e_2: $\delta(e_1, a_1) = e_2$. Learning agent A_L performs an action a_2 (generates a recommendation) after analyzing student feedback, which changes the state of the environment to e_3 (all reviews are collected): $\delta(e_2, a_2) = e_3$.

This model describes a multiagent system through sets of agents, actions, and environment states. The transition function allows you to track how the actions of agents affect the environment, and temporal logic allows you to control the change of states over time. The function of the learning agent $F_L : C \times R \to Act(A_L)$ generates recommendations based on student feedback, where C — courses, R — reviews. The behavior specification of a polling agent is described by the expression $F_Q : U \to Q$, where U – system users, and Q – Questionnaires. The agent sends questionnaires to users and collects responses:

$$G\big(F_Q(u) = q\big), u \in U, q \in Q.$$

For the system as a whole, optimization of the learning process corresponds to

$$G(\forall s \in S, a \in Act(A_S) : \delta(e_i, a_j) = e_{i+1}),$$

which means that the student's actions always lead to changes in the environment that contribute to better results.

The survey agent always helps to improve communication by collecting responses:

$$G(\forall a \in Act(A_Q) : \delta(e_i, a_j) = e_{i+1}).$$

Assistance in achieving results looks like this:

$$F(\forall a \in Act(A_L): \delta\delta(e_i, a_j) = e_{i+1}),$$

i.e., the recommendations of the learning agent ultimately lead to better results for students. These formulas formally describe a multi-agent decision support system, defining its structure, agent behavior, actions, and interactions with the environment through sets and logic.

3.2 Application of temporal logic

Temporal logic is important for analyzing and managing temporal aspects in systems where various events and processes occur over time. Therefore, we will create a formal specification of agents using temporal logic for decision making in distance learning. This approach allows us to create accurate and systematic models of agents and their behavior in a virtual learning environment. The formalization of agents and their behavior allows you to ensure the quality of learning and optimize the process, which is especially important in distance learning. Formulas in temporal logic express relationships between events, states, or properties of objects at different points in time. To create formulas, we use:

- Logical operators. Temporal logic uses logical operators such as AND, OR, NOT, IMPLIES, EQUIV to build complex expressions from basic temporal or logical statements.
- Temporal operators. Temporal logic has its own temporal operators, such as "Next" (X(r) – in the next step: agents automatically adapt the curriculum for the student at the next stage based on the analysis results), "Until" (U(p, q) – “until": students complete current learning tasks until an improvement in their knowledge level is achieved), "Future " (F(q) or <>– in the future: for each student there is a point in time when the system will offer optimized recommendations to improve academic performance.), " Globally" (G(p) or [ ] – globally (always): if p is true in all future states of the system, then each student receives educational materials according to his/her level of knowledge), which are used to express relationships between events at different points in time, operator "Release" (R) indicates that the event it wraps will be true until the second event occurs.
- Parameterization. Formulas can contain parameters that depend on a specific context or system conditions. This allows you to generalize formulas and ensure their use in different scenarios.
- Quantifiers. Temporal logic can use quantifiers such as "For All" (∀) and "Exists" (∃) to express general or existing relationships between events.
- Event and state identifiers. Formulas can contain identifiers of events, states, or object properties that are used to describe temporal aspects and their relationships with system states.
- Actions and events. Formulas can include a description of actions, events, or observations that occur in the system and their impact on the states and properties of the system over time.

- Relationships to other logical and mathematical systems. Formulas in temporal logic can use constructs and concepts from other logical and mathematical systems to further express temporal properties.

Temporal logic allows you to describe the evolution of the system state in time.

Let t — moment in time. Then formalizing the situation when the learning process for a student s_i at any given time must comply with:

- specific learning material or task t_k, that he is working on,
- and a fixed level of knowledge l_u, that this student has at the moment

$$G(F_T(s_i) = t \wedge F_L(s_i) = l_u), \quad (4)$$

$$F(F_R(s_i) = r_\varphi)), \quad (5)$$

$$X(F_T(s_i) = t_k{}' \wedge F_R(s_i) = r_\varphi{}'), \quad (6)$$

$$(F_T(s_i) = t_k) \cup F_L(s_i) = l_u{}'), \quad (7)$$

where $t_k{}'$ – new training materials, a $r_\varphi{}'$ – new recommendations, $l_u{}' > l_u$.

Expression (4) means that always (at each stage or at each moment in time):

- student s_i a specific task or learning material is assigned t_k,
- and at the same time the level of knowledge of this student s_i is equal to l_u.

This formalizes the situation when the educational process for a student s_i at any given time must comply with:

- specific learning material or task t_k, that he is working on,
- and a fixed level of knowledge l_u, that this student has at the moment.

This logical formula can be used to check the correctness of the learning process in the system: the student should always know his or her current level of knowledge and the material he or she is studying. Expression (5) means that in the future (at some point) the student s_i achieve the result r_φ. In other words, in the future, the condition must be met that the student's result s_i will be equal to r_φ. If it is a student s_i will be evaluated r_φ=85, the expression $F(F_R(s_i)= 85))$, means that in the future the student s_i will achieve a score of 85 points. This condition does not require immediate or constant fulfillment, but it must occur at some point in the future. That is, the temporal operator F sets the condition for achieving a certain result in the future, and the expression $F_R(s_i)= r_\varphi$ determines what kind of result the student s_i should achieve. The entry (6) means that at the next moment in time, the student s_i a new task will be assigned $t_k{}'$, at the same time the student's result s_i will be equal to $r_\varphi{}'$. Function. $F_T(s_i) = t_k$ reflects the student s_i to a learning task or material $t_k{}'$ in the system. That is, a student s_i at the next time a new task will be assigned $t_k{}'$.

Function. $F_R(s_i) = r_\varphi'$ determines the student's result s_i у системі. У наступний момент часу результат студента s_i will be equal to r_φ', де r_φ' – a certain level of performance or score. For example, if a student s_i is currently performing the following tasks t_k, and its result is currently equal to r_φ. Recording $X(F_T(s_i) = t_k' \wedge F_R(s_i) = 85)$ means that at the next point in time, the student s_i a new task will be assigned t_k', and at the same time its result will be 85 points. Expression (7) means that either the student s_i a task has been assigned t_k, or the student's level of knowledge s_i is equal to l_u', or both conditions can be met simultaneously. For example, if a student s_i a task has been assigned t_3 (e.g., the third assignment in the course), and its level of knowledge may be equal to l_5 (level five). Expression. ($(F_T(s_i) = t_3) \cup F_L(s_i) = l_5')$, means that either the student is assigned a task t_3, or his level of knowledge is l5 (or both statements are true at the same time). One approach is to use a formal language such as LTL to define temporal logic rules. LTL expressions include time operators that allow you to express conditions and state changes over time. A rule for detecting misbehavior over time, i.e., if there is a moment in time when misbehavior is detected, then the agent should respond:

LTL expression: <> (Unlawful_Behavior).

A rule to check for misbehavior at a certain frequency, i.e., if misbehavior is detected during each "Period" of steps, then the agent should react:

LTL expression: <> (Unlawful_Behavior) ∪ (every_n_steps(Period)).

The rule for the appeal period, i.e.: "There is a point in time when the breach notification was sent, and this event is true until the appeal response is received."

LTL expression: <> (Sent_Notification_of_Violation) R <> (Received_Appeal_Response).

A rule to prohibit the use of aids, i.e. if a student has used aids, the agent must block the test:

LTL expression: <> (Used_Auxiliary_Tools) → <> (Block_Test).

The rule for applying warnings, i.e. if a violation is detected, the agent must send a warning:

LTL expression: <> (Violation_Detected) → <> (Send_Warning).

For the formal specification of agent-based decision making in distance learning, we use the formal description language TLA+ (Temporal Logic of Actions). TLA+ allows modeling and formalizing systems with temporal aspects, and helps to express and verify the properties of specific systems, including distributed systems and algorithms. This specification language allows you to formally define systems, taking into account their logic and dynamics. This helps to avoid misunderstandings and allows you to accurately define the expected behavior of the system.

The formal specification of an agent-oriented decision support system for distance learning is given by the following formulas:

Specification 1:
<> (Unlawful_Behavior)
<> (Unlawful_Behavior) U (every_n_steps(Period))
<> (Sent_Notification_of_Violation) R <> (Received_Appeal_Response)
<> (Used_Auxiliary_Tools) → <> (Block_Test)
<> (Violation_Detected) → <> (Send_Warning)
G (time = 0 → (testQuestions = <<q1, q2, q3>> ∧ studentAnswers = <<>> ∧ teacherRecommendations = <<>> ∧ violations = <<>> ∧ appeals = <<>>))
G (time < MaxTime → (∃ student ∈ DOMAIN(studentAnswers): X (studentAnswers[student][time'] = CHOOSE answer ∈ testQuestions)))
G (time = MaxTime → X GenerateTeacherRecommendations)
G (GenerateTeacherRecommendations → (CheckForViolations ∧ CheckForAppeals ∧ CheckForWarning))
G (CheckForViolations → (violations' = ...))
G (CheckForAppeals → (appeals' = ...))
G (CheckForWarning → (IF violations' > WarningThreshold THEN teacherRecommendations' = AppendWarning(teacherRecommendations) ELSE teacherRecommendations' = teacherRecommendations ENDIF))
G (AppendWarning(recommendations) → (teacherRecommendations' = Append(recommendations, "Warning")))
G (AppealDecision → (∃ appeal ∈ appeals: (IsValidAppeal(appeal) ∧ ProcessAppeal(appeal))))
G (WarningDecision → (IF violations > WarningThreshold THEN teacherRecommendations' = AppendWarning(teacherRecommendations) ELSE teacherRecommendations' = teacherRecommendations ENDIF))

3.3. Verification of multi-agent systems in education

To verify the correctness of concurrent software models, a generic tool called SPIN is used, which is aimed at efficiently verifying multithreaded software. Spin has been used to track down logical errors in the design of distributed systems such as operating systems, data communication protocols, switching systems, parallel algorithms, railroad signaling protocols, spacecraft control software, nuclear power plants, and more. The tool checks the logical sequence of the specification and interlock reports, competition conditions, different types of incompleteness, and unreasonable assumptions about the relative speed of processes. The properties to be checked are expressed as linear time logic (LTL) formulas. That is, you can formally specify the expected behavior of the system and check its fulfillment. To verify the proposed model using SPIN, we define the main components of the system, including agents, their actions, possible states, and state transition functions in Promela:

Each agent is presented as a separate procedure proctype

- Agent actions are represented as conditional blocks if/else or do/od, where the agent changes its state depending on the conditions.

- Agent states are defined through variables of type mtype.
- Transitions between states are realized with the help of conditional structures.

Then the model of a multiagent system will be presented in the following form:

```
mtype = {idle, active, analyzing, communicating, done};// The set of agent states
mtype = {complete_task, pass_test, view_material, generate_recommendation, analyze_feedback, send_survey, collect_responses, process_data, detect_patterns}; // A set of agent actions
chan communication = [1] of {mtype}; // A channel for exchanging messages between agents
// Procedure for a student agent
proctype AgentStudent() {
    mtype state = idle; //Initial state of the student agent
    do
    :: (state == idle) -> state = complete_task;
    :: (state == complete_task) -> state = pass_test;
    :: (state == pass_test) -> state = view_material;
    :: (state == view_material) -> communication!done
    :: (state == done) -> break;
    od;
}
 // Procedure for a training agent
proctype AgentLearning() {
    mtype state = idle; // The initial state of the training agent
    do
    :: (state == idle) -> state = analyze_feedback;
    :: (state == analyze_feedback) -> state = generate_recommendation;
    :: (state == generate_recommendation) -> communication?done -> state = done;
    od;
}
// Procedure for a data analysis agent
proctype AgentDataAnalysis() {
    mtype state = idle; // The initial state of the data analysis agent
    do
    :: (state == idle) -> state = process_data;
    :: (state == process_data) -> state = detect_patterns;
    :: (state == detect_patterns) -> communication!done;
    :: (state == done) -> break;
    od;
}
// Procedure for a survey agent
proctype AgentSurvey() {
    mtype state = idle; // The initial state of a survey agent
    do
```

```
    :: (state == idle) -> state = send_survey;
    :: (state == send_survey) -> state = collect_responses;
    :: (state == collect_responses) -> communication!done;
    :: (state == done) -> break;
    od;
}
// Procedure for system agent
proctype AgentSystem() {
    mtype state = idle; // The initial state of the system agent
    do
    :: (state == idle) -> state = active; // System activation
    :: (state == active) -> state = done; // Termination of interactions with other agents
    od;
}
// Initialization of all agents
init {
    atomic {
        run AgentStudent();
        run AgentLearning();
        run AgentSurvey();
        run AgentDataAnalysis();
        run AgentSystem();
    }
}
```

States idle, active, analyzing, communicating, done represent different stages in the work of agents: waiting, active work, analysis, communication, and completion. Each agent performs its actions depending on its current state. The transitions between agent states are defined in conditional statements do/od, where each agent changes its state in accordance with the specified conditions. After reviewing the materials, the agent AgentStudent sends messages via the communication channel. Agent AgentLearning receives notifications from the channel after the agent completes tasks AgentStudent and exits the survey. When the collection of responses is complete, AgentSurvey sends a message through the channel. To simplify the system, we assume that AgentSystem remains unchanged, only activates the system and exits. Channel. communication is used to exchange states between agents. Agent AgentStudent and agent AgentSurvey send messages to the channel, and the agent AgentLearning receives them and reacts accordingly. We use this model to test the interaction of agents in SPIN to make sure that they perform their actions correctly and communicate with each other through the communication channel.

In the simulation mode, we can see how agents change their states and transmit messages through channels. For example, an agent AgentStudent completes its task and sends a message through the channel, and the agent AgentLearning receives this message and continues its work. After executing the simulation command: spin -p -g -l model.pml, we get the following result:

1: proc 0 (init) line 26 "model.pml" (state 1) [atomic {run AgentStudent();run AgentLearning();}]

2: proc 1 (AgentStudent) line 6 "model.pml" (state 1) [state == idle]

2: proc 1 (AgentStudent) line 6 "model.pml" (state 2) [state = active]

3: proc 1 (AgentStudent) line 7 "model.pml" (state 2) [communication!done]

3: proc 1 (AgentStudent) line 8 "model.pml" (state 3) [state == done]

4: proc 2 (AgentLearning) line 12 "model.pml" (state 1) [state == idle]

4: proc 2 (AgentLearning) line 12 "model.pml" (state 2) [state = analyzing]

5: proc 2 (AgentLearning) line 13 "model.pml" (state 2) [communication?done]

5: proc 2 (AgentLearning) line 14 "model.pml" (state 3) [state = communicating]

6: proc 2 (AgentLearning) line 15 "model.pml" (state 4) [state == done]

Thus, the process of proc 1 (AgentStudent) means that the AgentStudent agent starts in the idle state (line 2), moves to the active state (line 2); the AgentStudent agent sends a done message through the communication channel (line 3); the AgentStudent agent completes its work by moving to the done state (line 3). The process proc 2 (AgentLearning) means that the AgentLearning agent starts in the idle state (line 4), enters the analyzing state (line 4); the AgentLearning agent waits for a message through the channel. As soon as it receives a done message, it enters the communicating state (line 5). After that, the AgentLearning agent completes its work by switching to the done state (line 6). The analysis shows that the transmission of a message through the communication!done channel from the AgentStudent agent is successful, and the AgentLearning agent receives this message through communication?done. Both agents correctly change their states and complete their work, which is evident at the end of the simulation for both processes in the done state. This simulation result demonstrates the correctness of the interaction of agents through the communication channel and the execution of their actions.

LTL formulas make it possible to formally describe the behavioral properties of multi-agent systems and check their correctness using SPIN. For this purpose, the following operators are used [], <>, X, U and others, which are translated into Promela through the corresponding expressions ltl. (Table 1).

Table 1
Designation of specific LTL operators in keywords in Promela

Operators LTL	Promela keywords
[ ](p)	ltl always_p { [](p) }
< >(q)	ltl eventually_q { <>(q) }
X(r)	ltl next_r { X(r) }
p U q	ltl until_p_q { (p) U (q) }
p -> q	ltl implication_p_q { (p -> q) }
[](p -> <>q)	ltl always_p_eventually_q { [](p -> <>q) }

To verify the formal specification of an agent-based decision support system for distance learning using SPIN and the Promela language, we translate logical formulas based on linear temporal logic (LTL) (Specification 1) into the Promela code:

Step 1. Translate LTL formulas into Promela to check system properties:

```
ltl p1 { <> (Unlawful_Behavior) }
ltl p2 { <> (Unlawful_Behavior U every_n_steps(Period)) }
ltl p3 { <> (Sent_Notification_of_Violation) R <> (Received_Appeal_Response)
}
ltl p4 { <> (Used_Auxiliary_Tools) -> <> (Block_Test) }
ltl p5 { <> (Violation_Detected) -> <> (Send_Warning) }
…
```

Step 2. Creating a system model in Promela based on the specification of agents, their actions, states, and possible events (e.g., violations, test blocking, etc.).

```
mtype = {idle, active, analyzing, violation_detected, send_warning, block_test};
proctype AgentStudent() {
   mtype state = idle;
   do
   :: (state == idle) -> state = active;
   :: (state == active) -> // Actions like completing tasks, answering questions
   od;
}
proctype AgentLearning() {
   mtype state = idle;
   do
   :: (state == idle) -> state = analyzing;
   :: (state == analyzing) -> // Generate recommendations
   od;
}
// Define LTL formula for detecting violations and warnings
ltl p1 { <> (violation_detected) }
ltl p2 { violation_detected -> <> (send_warning) }
// Initialize system
init {
   atomic {
      run AgentStudent();
      run AgentLearning();
   }
}
```

Step 3. Generate a model for verification

Step 4. Compile the generated code

Step 5. Run the verification of each formula.

Step 6. Analyze the results.

For example, we check the property that guarantees that “sooner or later a violation will occur in the system,” i.e. Unlawful_Behavior: ltl p1 { <> (Unlawful_Behavior) }

This means that on any system execution path, the Unlawful_Behavior event must occur at least once in the future.

When checking this property, SPIN can return several different results:

- Successful verification, if the system always has at least one execution path on which the Unlawful_Behavior event occurs at least once, SPIN will confirm that the system meets the specified formula (Fig. 1):

```
Valid: The formula holds in all executions of the system.
```

***Figure 1**. Successful verification*

- Verification error, if SPIN finds an execution path where the Unlawful_Behavior event never occurs (for example, the system can run indefinitely without a violation occurring), SPIN will return an error message and we will get an execution trace showing exactly how this situation occurred (Fig. 2).

```
Error: The formula does not hold in all executions.
```

***Figure 2**. Verification error*

If the system loops, if the system enters an infinite loop and Unlawful_Behavior is never called, SPIN can determine this as an LTL formula violation. That is, in an infinite loop, the system never reaches a state where Unlawful_Behavior can occur.

If the verification fails, SPIN provides an execution trace that shows the sequence of states the system went through and the events that did or did not occur. The execution trace is presented as a sequence of events and states of the system, which allows us to recreate how the system moved between states and where the verified property (LTL formula) was violated.

The execution trace contains information about the values of variables, communication channels, process states, and events that occurred at each step:

```
Trace for assertion violation
  proc  0 (AgentStudent) line  10 "model.pml" (state 3)
  proc  1 (AgentLearning) line  15 "model.pml" (state 2)
  proc  2 (AgentDataAnalysis) line  20 "model.pml" (state 1)
  proc  3 (AgentSystem) line  25 "model.pml" (state 4)
  proc  4 (AgentSurvey) line  30 "model.pml" (state 2)
  step  0: proc  0 (AgentStudent)
    state change: state -> active
  step  1: proc  1 (AgentLearning)
    state change: state -> analyzing
  step  2: proc  0 (AgentStudent)
    state change: state -> done
  step  3: proc  2 (AgentDataAnalysis)
    state change: state -> done
  step  4: proc  3 (AgentSystem)
    state change: state -> done
  step  5: proc  1 (AgentLearning)
    state change: state -> communicating
  step  6: proc  4 (AgentSurvey)
    state change: state -> analyzing_survey
step  7: assertion violated in line 20 of "model.pml"
```

Each "step" represents one step of system execution. Steps include information about which process (agent) was active and what state changes occurred. For example, in step 0, the AgentStudent process went to the active state, and in step 2, the same agent changed its state to done.

Processes represent system agents or other independent parts of the program. Each process has its own number and name (for example, AgentStudent), as well as a line of code and the state of the process at the time of the trace. If the trace leads to a specification violation, SPIN indicates the line of code where the violation occurred (Figure 3):

```
assertion violated in line 20 of "model.pml"
```

***Figure 3**. Property violation on line 20 of the code*

Thus, specification verification using SPIN and Promela ensures high reliability and accuracy of agent-based system development, helps minimize risks, and improves the overall quality of the system, especially in the context of distance learning, where the requirements for process correctness are particularly high.

3.4. Recommendation subsystem

To solve the urgent problems of each of the participants in the educational process, a subsystem for assessing student and teacher satisfaction has been developed. This approach allows for effective management of internal learning processes in an educational institution. The implementation of the subsystem for assessing student satisfaction using the online platform consists of the following stages.

Stage 1: Set the goal of increasing student and faculty satisfaction to a certain level. To measure satisfaction, we define key performance indicators (KPIs).

We develop questionnaires with questions that cover all aspects of using the platform. This questionnaire includes both closed and open-ended questions, which allows us to obtain both quantitative and qualitative data on student satisfaction with the online platform.

Stage 2. The online platform Google Forms was chosen for the surveys.

Questionnaire processing consists of collecting responses, processing data, analyzing results, and preparing a report.

Questionnaire processing algorithm:

Step 1. Placement of the questionnaire: Publish the questionnaire on an online platform or send it out by email. Set a deadline for submitting responses.

Step 2. Data collection: Saving all answers in a single database or spreadsheet.

Step3. Data preparation: Exporting the collected responses to a format convenient for processing (CSV or Excel). Checking the data for completeness and correctness.

Step 4. Categorization of answers: Dividing data into categories according to the questionnaire questions.

Step 5. Calculation of quantitative indicators: For closed-ended questions (scoring on a scale from 1 to 5), calculate mean, median, mode, and other statistics. Build graphs and charts to visualize the results.

Step 6. Calculation of quantitative indicators: For closed-ended questions (scoring on a scale from 1 to 5), calculate mean, median, mode, and other statistics. Build graphs and charts to visualize the results.

Step 7. Analysis of open responses: Analyzing open-ended responses to identify key themes and trends. Build word clouds or other visual representations to help interpret the results.

Step 9. Interpretation of quantitative data: Identify key trends and problem areas based on the calculated statistical indicators. Comparison of results between different groups (e.g., among students by age, course, etc.).

Step 9. Interpretation of qualitative data: Identification of the main topics and problems mentioned by participants in open-ended responses. Assessment of the general mood and tone of the answers.

Step 10. Preparation of the report: Prepare a report that includes key findings from the data analysis. Include graphs, charts and other visuals to illustrate the results.

Step 11. Recommendations: Developing recommendations based on the results to improve the online platform. Identification of priority areas for implementation of changes.

Step 12. Communication of results: Presentation of the report to the stakeholders (school administration, teachers, technical support). Publishing a summary of the main results for participants.

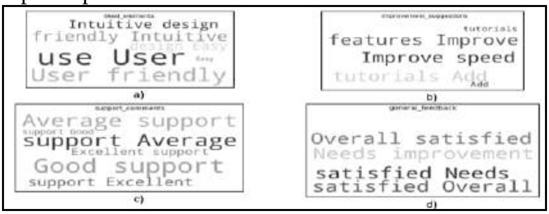

Figure 4. *a) The "liked_elements" cloud shows the most popular or frequently mentioned elements or features that user liked about the product. b) The "improvement_suggestions" cloud shows user recommendations for improvements in the product. c) The "support comments" cloud displays user comments about the quality of support. d) The "general_feedback" cloud shows the general feedback users have given to the product.*

Create the file survey_responses.csv, which will contain the survey data, containing columns that correspond to the survey questions. This file contains data from the survey, where each row represents the response of one survey participant to different questions. Each column corresponds to a specific question or aspect that was rated by the participants on a scale from 1 to 5, where 1 is very bad and 5 is very good. These graphs help to understand how users perceive a product or service and identify possible areas for improvement. In this case, qualitative data processing graphs are word clouds that visualize the most common words or phrases found in user comments. Each of these graphs corresponds to a specific question in the questionnaire (Fig. 4). This data will be used to analyze user satisfaction and identify areas for product or service improvement (Fig. 5).

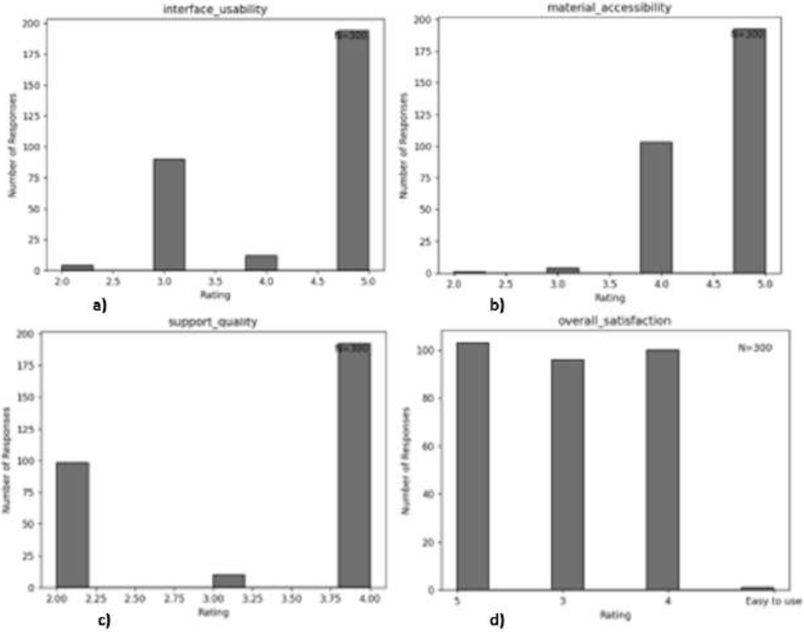

Figure 5. *a) The interface_usability graph shows the distribution of user ratings on the interface usability scale. b) The material_accessibility graph shows the distribution of material accessibility scores. c) The support_quality graph shows the distribution of support quality scores. d) The overall_satisfaction graph shows the distribution of overall user satisfaction with the product or service*

Based on the data analysis, the following recommendations were made for teachers:

- Improve support for technical and methodological issues – if the average value for this criterion is less than 4, technical and methodological support for teachers should be improved.
- Enhance opportunities for interactive interaction with students - if the average value for this criterion is less than 4, you need to increase opportunities for interactive interaction with students.
- Provide more options for professional development – if the average value for this criterion is less than 4, more opportunities for professional development of teachers should be provided.

Based on the analysis of the results of the survey of teachers on their satisfaction with the use of the online platform, several main areas for improvement can be identified:

1. Increase the efficiency of using electronic resources for teaching.

According to the survey, teachers evaluate the effectiveness of the use of electronic resources for teaching with an average of 4 out of 5 points. This shows overall satisfaction, but also indicates that there is room for improvement.

Recommendations: Simplify navigation and access to materials. Add the ability to create interactive learning materials (e.g., interactive presentations, video tutorials with interactive elements). Introduce new test formats and automatic tools for assessing students' knowledge.

2. Support for teachers in the use of e-learning systems.

Technical and methodological support received an average score of 4 out of 5, which indicates a sufficient level of satisfaction, but also points to the need for improvement.

Recommendations: Provide round-the-clock technical support through various channels (chat, phone, email). Regularly conduct trainings on how to use the platform, including new features and best practices. Provide an opportunity to order individual consultations to solve specific problems.

3. Expanding opportunities for interactive interaction with students.

Opportunities for interactive interaction with students are rated 4 out of 5, which shows the need for additional features to improve interaction.

Recommendations: Introduce additional tools for video conferencing, interactive forums and chats. Expand feedback opportunities, for example, by adding the ability to conduct surveys and questionnaires in real time. Introduce gamification elements to increase student motivation.

4. Assessment of opportunities for professional development of teachers.

Opportunities for professional development through the use of e-learning systems received an average score of 4 out of 5, which indicates the need to expand such opportunities.

Recommendations: Introduce professional development programs that include courses, webinars, and the possibility of obtaining certificates. Create a platform for the exchange of experience and best practices among teachers. Introduce a system of incentives for participation in professional development programs. The results of the survey show the overall satisfaction of teachers with various aspects of e-learning, as the average scores are at the level of 4 or higher.

4. Conclusion

The article proposes an agent-based method for improving the efficiency of e-learning, which consists of the following stages:

Stage 1: Analysis and preparation. The main goals of the system are identified: improving learning efficiency, automating processes, increasing student and teacher satisfaction. Existing systems and limitations were analyzed, and suitable technologies for agent-based systems were explored.

Stage 2. System design. The main components of the system were identified: a video surveillance subsystem, a multi-agent system, and a decision-making system. The architecture of interaction between the components was designed. Agents' roles and functions (data collection, analysis, recommendations) were established, and their interaction algorithms were formalized.

Stage 3. Verification. Formal confirmation of the correctness of the specification of the agent-based decision support system for distance learning was conducted using SPIN and the Promela language, which provides confidence that the system will work as expected, especially for critical properties such as preventing violations, providing timely feedback, and guaranteeing the accuracy of recommendations. Using SPIN allowed us to test different scenarios of agents' operation and their interaction at the design stage, which helps to avoid mistakes during the development phase or during system operation. The correctness of agent interactions was checked.

Stage 4. Development and implementation. Program modules for agents were developed, a video surveillance subsystem implemented, and both functional and integration testing were performed.

This method automates monitoring, analysis, and decision-making, improving education quality and satisfaction. Verification ensures system reliability and security. Simulation results show agents interact correctly and fulfill their tasks.

5. References

[1] Graesser, C. Arthur, Hu Xiangen, and Robert Sottilare, “Intelligent tutoring systems,” International handbook of the learning sciences. Routledge, 2018. pp.246-255.
[2] Nouri Jalal, “Editorial of the First Issue of the International Journal of Learning Analytics and Artificial Intelligence for Education,” International journal of learning analytics and artificial intelligence for education 1.1, 2019. pp. 4-7.
[3] F. Ouyang, P. Jiao, “Artificial intelligence in education: The three paradigms,” Computers and Education: Artificial Intelligence, 2, 2021,100020.
[4] M. Cukurova, C. Kent, R. Luckin, “Artificial intelligence and multimodal data in the service of human decision-making: A case study in debate tutoring,” British Journal of Educational Technology, 50 (6), 2019, pp. 3032-3046.
[5] C. Campbell, “The three paradigms of Artificial intelligence in secondary school education,” Computers and Education: Artificial Intelligence, 2, 2022, 100020.
[6] Shahbazi, Zeinab, and Yung-Cheol Byun, “Agent-based recommendation in E-learning environment using knowledge discovery and machine learning approaches,” Mathematics 10.7, 2022, 1192.
[7] Krouska, C. Troussas, M. Virvou, “SN-Learning: An exploratory study beyond e-learning and evaluation of its applications using EV-SNL framework,” J. Comput. Assist. Learn. 2019, 35, pp. 168–177.

[8] M.S. Rafiq, X. Jianshe, M. Arif, P. Barra, “Intelligent query optimization and course recommendation during online lectures in E-learning system,” J. Ambient. Intell. Humaniz. Comput. 2021, 12, pp. 10375–10394.
[9] Mbipom, S. Craw, S. Massie, “Improving e-learning recommendation by using background knowledge,” Expert Syst. 2021, 38, e12265.
[10] Y.R. Shrestha, S. M. Ben-Menahem, G. Von Krogh, “Organizational decision-making structures in the age of artificial intelligence,” California Management Review, 61(4), 2019, pp. 66-83.
[11] R. X. Ding, I. Palomares, X. Wang, G. R. Yang, B. Liu, Y. Dong, & F. Herrera, “Large-Scale decision-making: Characterization, taxonomy, challenges and future directions from an Artificial Intelligence and applications perspective,” Information fusion, 59, 2020, pp. 84-102.
[12] E. Carrillo, S. Yeotikar, S. Nayak, M. K. M. Jaffar, S. Azarm, J. W. Herrmann, H. Xu, “Communication-aware multi-agent metareasoning for decentralized task allocation”, IEEE Access, 2021, 9, pp. 98712-98730.
[13] Abate, J. Gutierrez, L. Hammond, P. Harrenstein, M. Kwiatkowska, M. Najib, M. Wooldridge, “Rational verification: game-theoretic verification of multi-agent systems”, Applied Intelligence, 2021. 51(9), pp. 6569-6584.
[14] R. Calegari, G. Ciatto, V. Mascardi, et al., “Logic-based technologies for multi-agent systems: a systematic literature review”, Auton Agent Multi-Agent Syst 35, 1, 2021. https://doi.org/10.1007/s10458-020-09478-3
[15] Nicolas Markey. Temporal Logics for Multi-Agent Systems (Invited Talk). In 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017). Leibniz International Proceedings in Informatics (LIPIcs), Volume 83, pp. 84:1-84:3, Schloss Dagstuhl - Leibniz-Zentrum für Informatik. 2017 https://doi.org/10.4230/LIPIcs.MFCS.2017.84
[16] Rybakov, V. V. Temporal Multi-Agents’ Logics Modelling Dynamic Temporal Accessibility Relations. Procedia Computer Science, 207, 2022, pp. 475-482.
[17] Hammond, L., Abate, A., Gutierrez, J., & Wooldridge, M. Multi-agent reinforcement learning with temporal logic specifications. arXiv preprint arXiv:2102.00582, 2021.
[18] Lamport, L. The temporal logic of actions. ACM Transactions on Programming Languages and Systems (TOPLAS), 16(3), 1994, pp. 872-923.
[19] Axak N., Tatarnykov A., The Behavior Model of the Computer User, IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT), 2022, pp. 458-461.
[20] Axak N., Kushnaryov M., Tatarnykov A., The Agent-Based Learning Platform. ICST-2023: XI International Scientific and Practical Conference “Information Control Systems and Technologies”, September 21-23, 2023, Odessa, Ukraine, Vol-3513, pp. 263-275.
[21] Axak N. G., Gryzun L. E., Shcherbakov O. V. [and others]; in general ed. V. S. Ponomarenko, “Modern information technologies and systems”: monograph /— Electron. text data (22.9 MB). — Kharkiv: HNEU named after S. Kuznetsia, 2022. — pp. 7–34. URL: http://repository.hneu.edu.ua/handle/123456789/29233. (in Ukrainian).

АГЕНТНИЙ ПІДХІД ДО ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ЕЛЕКТРОННОГО НАВЧАННЯ

Dr.Sci. Н. Аксак ORCID: 0000-0001-8372-8432
Харківський національний університет радіоелектроніки, Україна
E-mail: nataliia.axak@nure.ua
Ph.D. М. Кушнарев ORCID: 0000-0002-3772-3195
Харківський національний університет радіоелектроніки, Україна
E-mail: maksym.kushnarov@nure.ua
А. Татарников ORCID: 0000-0002-1632-8188
Харківський національний університет радіоелектроніки, Україна
E-mail: andrii.tatarnykov@nure.ua

Анотація. *У роботі розглядаються такі проблеми сучасного дистанційного навчання, як індивідуалізація навчального процесу. Обґрунтовано необхідність вдосконалення методів підтримки дистанційного навчання для підвищення якості освіти. Дослідження спрямоване на підвищення якості освіти та організації навчального процесу за дистанційної форми навчання. Запропоновано комплекс взаємопов'язаних завдань, розв'язання яких полягає в організації ефективної взаємодії викладача та системи управління навчанням (LMS) за допомогою агентного підходу. Основним результатом, який визначає новизну роботи, є формалізація та інтеграція таких процесів: (i) генерування в режимі реального часу пропозицій викладачеві щодо контролю за виконанням завдань під час іспитів чи електронного тестування; (ii) моніторинг навчання студентів протягом семестру з можливістю зміни траєкторії навчання; (iii) моніторинг присутності батьків на онлайн-класах; (iv) створення рекомендацій для керівництва та інших зацікавлених сторін щодо покращення онлайн-навчання. Також представлено перевірку формальної специфікації системи прийняття рішень на основі агентів для дистанційного навчання з використанням інструменту SPIN та мови моделювання Promela. Верифікація здійснюється за формулами лінійної часової логіки (ЛТЛ), що дозволяє перевірити коректність взаємодії між агентами та виконання ключових властивостей системи, таких як своєчасна обробка даних, реагування на порушення виконання студентами тестування. правила та рекомендації вчителів. Моделювання прототипу запропонованої системи підтверджує ефективність її використання як засобу дослідження організації навчального процесу. Він показує, як агенти можуть допомогти збирати й аналізувати дані про те, наскільки ефективно використовуються електронні ресурси для навчання студентів і викладачів.*

Ключові слова: *система управління навчанням, моніторинг, навчальний процес, агенти, хмарні обчислення, специфікації часової логіки розподіленої системи, часова логіка змінного часу, верифікація*