

НАЦІОНАЛЬНИЙ АЕРОКОСМІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АЕРОКОСМІЧНИЙ УНІВЕРСИТЕТ
«ХАРКІВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова праця
на правах рукопису

МАЛЮГА АРТУР ІВАНОВИЧ

УДК 004.94:004.65:629.3.027

ДИСЕРТАЦІЯ

МОДЕЛЬ ТА МЕТОДИ ПЕРСОНІФІКОВАНОЇ АДАПТАЦІЇ ПРОЦЕСУ
ПІДГОТОВКИ ВОДІЇВ В НАВЧАЛЬНИХ ПРОГРАМНИХ СИСТЕМАХ
АВТОШКІЛ

121 Інженерія програмного забезпечення

12 Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень.

Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ А.І. Малюга

Науковий керівник: д.т.н., професор Шостак Ігор Володимирович

Харків – 2026

АНОТАЦІЯ

Малюга А. І. Модель та методи персоніфікованої адаптації процесу підготовки водіїв в навчальних програмних системах автошкіл. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 121 – Інженерія програмного забезпечення Національний аерокосмічний університет «Харківський авіаційний інститут», Харків, 2026.

Дисертацію присвячено розробці моделі та методів та персоніфікованої адаптації в програмних системах навчання в автошколі для підвищення ефективності підготовки водіїв з урахуванням як рівня підготовки користувача, так і його поточного когнітивного стану.

Метою дисертаційного дослідження є розробка моделі та методів персоніфікованої адаптації процесу підготовки водіїв на базі гібридної архітектури програмної системи з використанням формалізованого представлення профілю знань та поточного стану користувача для підвищення ефективності навчання в програмних системах автошкіл.

В дисертаційному дослідженні вирішуються наступні завдання:

1. Аналіз стану проблеми адаптації та персоніфікації процесу навчання водіїв у автошколі.
2. Розробка моделі гібридної програмної архітектури з периферійними обчисленнями для персоніфікованої адаптації процесу підготовки в автошколі.
3. Розробка методів побудови вектору профілю користувача та вектору поточного стану користувача навчальної програмної системи.
4. Розробка методу персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача.
5. Розробка методу верифікації процесу вибору навчальних модулів з використанням технології інтелектуального аналізу процесів.
6. Реалізація гібридної програмної архітектури та експериментальна перевірка розроблених методів персоніфікованої адаптації процесу навчання.

Об'єктом дослідження є процес персоніфікованої адаптації в програмних системах навчання в автошколі.

Предметом дослідження є модель та методи персоніфікованої адаптації в програмних системах навчання, що враховують рівень підготовки та поточний стан користувача.

В дисертації використано методи комп'ютеризованого адаптивного тестування, машинного навчання, підтримки прийняття рішень, інтелектуального аналізу процесів та архітектурного проектування гібридних програмних систем. Метод формування вектору профілю користувача навчальної програмної системи використовує методи комп'ютеризованого адаптивного тестування та моделі адаптивних навчальних систем. Метод побудови вектору поточного стану користувача в навчальній програмній системі використовує моделі машинного навчання для оцінки варіабельності серцевого ритму. Метод персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача використовує методи підтримки прийняття рішень та теорію контекстно-адаптивних програмних систем. Метод верифікації процесу вибору навчальних модулів ґрунтується на методах інтелектуального аналізу процесів. Модель програмної архітектури з периферійними обчисленнями для персоніфікованої адаптації процесу підготовки в автошколі розроблено із використанням гібридних програмних архітектур.

Науковим завданням дисертаційного дослідження є розробка моделі та методів та персоніфікованої адаптації в програмних системах навчання в автошколі для підвищення ефективності підготовки водіїв з урахуванням як рівня підготовки користувача, так і його поточного когнітивного стану.

Наукова новизна отриманих в дисертаційному дослідженні результатів.

1. Вперше запропоновано модель гібридної програмної архітектури з периферійними обчисленнями системи навчання в автошколі, яка містить клієнтський рівень для візуалізації дорожнього середовища, периферійний рівень для диспетчеризації персоніфікованого вибору навчальних завдань та

мікросервісний рівень для верифікації вибору навчальних завдань, що створює умови для персоніфікованої адаптації процесу навчання в автошколі.

2. Удосконалено методи побудови векторів профілю та поточного стану користувача навчальної програмної системи, які на відміну від існуючих поєднують опис як рівня знань користувача з використанням психометричних підходів, так і його поточного психоемоційного стану на основі оцінки варіабельності серцевого ритму, що створює умови для автоматизованої диспетчеризації наступних навчальних модулів з урахуванням не лише рівня підготовки користувача, а й рівня його поточного когнітивного навантаження.

3. Вперше запропоновано метод персоніфікованої адаптації процесу навчання шляхом вибору навчальних модулів на основі векторів профілю та стану користувача та з урахуванням ступеню новизни завдань та попередньої фільтрації модулів за рівнем складності, що дає можливість у автоматизованому режимі підібрати навчальний сценарій згідно рівня підготовки та поточного когнітивного навантаження користувача навчальної програмної системи.

4. Набув подальшого розвитку метод верифікації процесу підготовки водіїв в навчальних програмних системах автошкіл, який на відміну від існуючих містить етапи побудови моделей цільового та фактичного процесів персоніфікованої адаптації навчання та виявлення порушень адаптивної логіки на основі порівняння цільової та фактичної моделей, що забезпечує умови для удосконалення правил персоніфікованої адаптації у процесі навчання з урахуванням рівня латентних знань користувачів.

Практичне значення отриманих нових теоретичних результатів полягає у розробці гібридної програмної архітектури та модулів підтримки персоніфікованої адаптації навчального процесу з урахуванням рівня знань та поточного стану користувача програмної системи навчання в автошколах, що створює умови для підвищення ефективності навчання не лише з урахуванням складності опанованих користувачем навчальних сценаріїв, а й з урахуванням його поточного психоемоційного стану в умовах виникнення когнітивного перенавантаження під час навчання.

Отримані в дисертаційному дослідженні результати опубліковано в 10 наукових працях, в тому числі 5 статей у періодичних фахових наукових виданнях України з технічних наук категорії «Б», 5 тез доповідей на міжнародних конференціях.

Ключові слова: Індустрія 4.0, знання, моніторингова система, генеративний штучний інтелект, діагностична модель, тестовий сценарій, інформаційна система, оптимізація, тестова модель, децентралізоване навчання та виконання, агентне імітаційне моделювання, проїзд регульованого світлофорного перехрестя, приватна хмара, експериментальні дослідження, результати впровадження.

Перелік публікацій здобувача за темою дисертаційної роботи

1. Малюга А. І. Концептуальні засади дослідження впровадження технологій віртуальної реальності у навчальний процес автошкіл // *Вчені записки ТНУ імені В. І. Вернадського. Серія: Технічні науки*. – 2024. – Т. 35 (74), № 1. – С. 214–221. – DOI: 10.32782/2663-5941/2024.1.1/33.

2. Малюга А. І. Гібридна програмна архітектура з периферійними обчисленнями для адаптивних VR-систем навчання водінню з біометричним зворотним зв'язком // *Авіаційно-космічна техніка і технологія*. – 2025. – № 5. – С. 96–111. – DOI: 10.32620/aktt.2025.5.09.

3. Малюга А. І. Методи побудови адаптивного програмного забезпечення навчальних систем // *Авіаційно-космічна техніка і технологія*. – 2025. – № 6. – С. 113–127. – DOI: 10.32620/aktt.2025.6.11.

4. Малюга А. І. Метод онлайн-диспетчеризації тематичних модулів в адаптивних навчальних системах із використанням векторів індивідуального профілю та поточного стану користувача // *Авіаційно-космічна техніка і технологія*. – 2026. – № 1. – С. 123–132. – DOI: 10.32620/aktt.2026.1.11.

5. Малюга А. І., Соколова Є. В. Метод верифікації адаптивних програмних систем з використанням технології process mining // *Відкриті інформаційні та*

комп'ютерні інтегровані технології. – 2026. – № 107. – С. 257–263. – DOI: 10.32620/oikit.2026.107.17.

6. Малюга А. І. Шляхи підвищення ефективності процесу навчання навичкам водіння автомобільної та спеціальної техніки за рахунок впровадження спеціалізованих програмних засобів // *Проблеми інформатизації: тези доповідей дванадцятої міжнародної науково-технічної конференції (21–22 листопада 2024 р.)*. – Баку – Харків – Бельсько-Бяла, 2024. – Т. 2, секція 4. – С. 28.

7. Малюга А. І. Особливості застосування штучних нейронних мереж у тренажерах для підготовки водіїв автомобільної та спеціальної техніки // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей п'ятнадцятої міжнародної науково-технічної конференції (24–25 квітня 2025 р.)*. – Баку – Харків – Жиліна, 2025. – Т. 1, секції 1, 5. – С. 28.

8. Малюга А. І. Моделювання процесу обробки біометричних даних у середовищі віртуальної реальності при адаптивному навчанні водіїв // *Проблеми інформатизації: тези доповідей тринадцятої міжнародної науково-технічної конференції (27–28 листопада 2025 р.)*. – Баку – Харків – Бельсько-Бяла, 2025. – Т. 1, секції 1, 2. – С. 36.

9. Малюга А. І. Верифікація адаптивних програмних систем, що змінюють поведінку у відповідності до поточного психофізіологічного стану користувача // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей шістнадцятої міжнародної науково-технічної конференції (23–24 квітня 2026 р.)*. – Баку – Харків – Жиліна, 2026. – Т. 1, секція 2. – С. 43.

10. Шостак І. В., Малюга А. І. Методичні і програмні засоби персоніфікованої адаптації навчального процесу в автошколі // *Інтегровані інтелектуальні робототехнічні комплекси (ІРТК-2026): матеріали дев'ятнадцятої міжнародної науково-практичної конференції (19–20 травня 2026 р.)*. – Київ: НАУ, 2026. – С. 557-558.

ABSTRACT

Maliuha A. I. Model and Methods of Personalized Adaptation of the Driver Training Process in Educational Software Systems of Driving Schools. – Qualifying scientific work in manuscript form.

Dissertation submitted for the degree of Doctor of Philosophy, specialty 121 – Software Engineering. National Aerospace University "Kharkiv Aviation Institute", Kharkiv, 2026.

The dissertation is devoted to the development of a model and methods of personalized adaptation in driving school educational software systems aimed at improving the effectiveness of driver training, taking into account both the user's level of preparation and their current cognitive state.

The objective of the dissertation research is to develop a model and methods of personalized adaptation of the driver training process based on a hybrid software architecture, using a formalized representation of the user's knowledge profile and current state to improve learning effectiveness in driving school software systems.

The dissertation research addresses the following tasks:

1. Analysis of the state of the problem of adaptation and personalization of the driver training process in driving schools.
2. Development of a hybrid software architecture model with edge computing for personalized adaptation of the training process in a driving school.
3. Development of methods for constructing the user profile vector and the current user state vector of the educational software system.
4. Development of a method for personalized adaptation of the learning module selection process based on user profile and state vectors.
5. Development of a method for verifying the learning module selection process using process mining technology.

6. Implementation of the hybrid software architecture and experimental validation of the developed methods of personalized adaptation of the learning process.

The object of the research is the process of personalized adaptation in driving school educational software systems.

The subject of the research is the model and methods of personalized adaptation in educational software systems that account for the user's level of preparation and current state.

The dissertation employs methods of computerized adaptive testing, machine learning, decision support, process mining, and architectural design of hybrid software systems. The method for constructing the user profile vector of the educational software system applies computerized adaptive testing techniques and adaptive learning system models. The method for constructing the current user state vector in the educational software system employs machine learning models for heart rate variability assessment. The method of personalized adaptation of the learning module selection process based on user profile and state vectors applies decision support methods and the theory of context-adaptive software systems. The method for verifying the learning module selection process is grounded in process mining methods. The hybrid edge-computing software architecture model for personalized adaptation of the driver training process was developed using hybrid software architecture principles.

The scientific objective of the dissertation is to develop a model and methods of personalized adaptation in driving school educational software systems to improve the effectiveness of driver training, taking into account both the user's level of preparation and their current cognitive state.

Scientific novelty of the results obtained in the dissertation research.

1. For the first time, a hybrid edge-computing software architecture model for a driving school learning system is proposed, comprising a client tier for road environment visualization, an edge tier for dispatching personalized selection of learning tasks, and a microservice tier for verifying learning task selection — thereby establishing conditions for personalized adaptation of the driving school training process.

2. The methods for constructing user profile and current state vectors of the educational software system have been improved; unlike existing approaches, they combine a description of the user's knowledge level using psychometric methods with an assessment of their current psycho-emotional state based on heart rate variability

analysis – enabling automated dispatching of subsequent learning modules that accounts not only for the user's level of preparation but also for their current cognitive load.

3. For the first time, a method of personalized adaptation of the learning process is proposed through the selection of learning modules based on user profile and state vectors, with consideration of task novelty and preliminary filtering of modules by difficulty level – enabling automated matching of a learning scenario to the user's level of preparation and current cognitive load in the educational software system.

4. The method for verifying the driver training process in driving school educational software systems has been further developed; unlike existing approaches, it incorporates stages for constructing models of the target and actual processes of personalized learning adaptation and for detecting violations of adaptive logic through comparison of the target and actual models — providing conditions for refining personalized adaptation rules during training with regard to users' latent knowledge levels.

The practical significance of the obtained theoretical results lies in the development of a hybrid software architecture and modules supporting personalized adaptation of the learning process that account for the user's knowledge level and current state in driving school software systems — establishing conditions for improving learning effectiveness not only in terms of the complexity of learning scenarios mastered by the user, but also with regard to their current psycho-emotional state under conditions of cognitive overload during training.

The results obtained in the dissertation research are published in 10 scientific works, including 5 articles in peer-reviewed professional scientific journals of Ukraine in technical sciences, category "B", and 5 conference abstracts presented at international conferences.

Keywords: Industry 4.0, knowledge, monitoring system, generative artificial intelligence, diagnostic model, test scenario, information system, optimization, test model, decentralized learning and execution, agent-based simulation, traffic light intersection, private cloud, experimental research, implementation results.

List of publications

1. Малюга А. І. Концептуальні засади дослідження впровадження технологій віртуальної реальності у навчальний процес автошкіл // *Вчені записки ТНУ імені В. І. Вернадського. Серія: Технічні науки*. – 2024. – Т. 35 (74), № 1. – С. 214–221. – DOI: 10.32782/2663-5941/2024.1.1/33.
2. Малюга А. І. Гібридна програмна архітектура з периферійними обчисленнями для адаптивних VR-систем навчання водінню з біометричним зворотним зв'язком // *Авіаційно-космічна техніка і технологія*. – 2025. – № 5. – С. 96–111. – DOI: 10.32620/aktt.2025.5.09.
3. Малюга А. І. Методи побудови адаптивного програмного забезпечення навчальних систем // *Авіаційно-космічна техніка і технологія*. – 2025. – № 6. – С. 113–127. – DOI: 10.32620/aktt.2025.6.11.
4. Малюга А. І. Метод онлайн-диспетчеризації тематичних модулів в адаптивних навчальних системах із використанням векторів індивідуального профілю та поточного стану користувача // *Авіаційно-космічна техніка і технологія*. – 2026. – № 1. – С. 123–132. – DOI: 10.32620/aktt.2026.1.11.
5. Малюга А. І., Соколова Є. В. Метод верифікації адаптивних програмних систем з використанням технології process mining // *Відкриті інформаційні та комп'ютерні інтегровані технології*. – 2026. – № 107. – С. 257–263. – DOI: 10.32620/oikit.2026.107.17.
6. Малюга А. І. Шляхи підвищення ефективності процесу навчання навичкам водіння автомобільної та спеціальної техніки за рахунок впровадження спеціалізованих програмних засобів // *Проблеми інформатизації: тези доповідей дванадцятої міжнародної науково-технічної конференції (21–22 листопада 2024 р.)*. – Баку – Харків – Бельсько-Бяла, 2024. – Т. 2, секція 4. – С. 28.
7. Малюга А. І. Особливості застосування штучних нейронних мереж у тренажерах для підготовки водіїв автомобільної та спеціальної техніки // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей п'ятнадцятої міжнародної науково-технічної*

конференції (24–25 квітня 2025 р.). – Баку – Харків – Жиліна, 2025. – Т. 1, секції 1, 5. – С. 28.

8. Малюга А. І. Моделювання процесу обробки біометричних даних у середовищі віртуальної реальності при адаптивному навчанні водіїв // *Проблеми інформатизації: тези доповідей тринадцятої міжнародної науково-технічної конференції (27–28 листопада 2025 р.).* – Баку – Харків – Бельсько-Бяла, 2025. – Т. 1, секції 1, 2. – С. 36.

9. Малюга А. І. Верифікація адаптивних програмних систем, що змінюють поведінку у відповідності до поточного психофізіологічного стану користувача // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей шістнадцятої міжнародної науково-технічної конференції (23–24 квітня 2026 р.).* – Баку – Харків – Жиліна, 2026. – Т. 1, секція 2. – С. 43.

10. Шостак І. В., Малюга А. І. Методичні і програмні засоби персоніфікованої адаптації навчального процесу в автошколі // *Інтегровані інтелектуальні робототехнічні комплекси (ІРТК-2026): матеріали дев'ятнадцятої міжнародної науково-практичної конференції (19–20 травня 2026 р.).* – Київ: НАУ, 2026. – С. 557-558.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	14
ВСТУП.....	15
1 АНАЛІЗ СТАНУ ПРОБЛЕМИ АДАПТАЦІЇ ТА ПЕРСОНІФІКАЦІЇ ПРОЦЕСУ НАВЧАННЯ ВОДІЇВ У АВТОШКОЛІ	20
1.1 Аналіз особливостей адаптивного процесу навчання водіїв в автошколах	20
1.2 Аналіз архітектур навчальних програмних систем	30
1.3 Дослідження підходів до персоніфікації навчання.....	41
1.4 Постановка задачі дослідження.....	52
Висновки до розділу 1	54
2 МОДЕЛІ ПІДТРИМКИ ПЕРСОНІФІКОВАНОГО АДАПТИВНОГО НАВЧАННЯ ВОДІЇВ У АВТОШКОЛАХ.....	55
2.1 Дескриптивне моделювання дорожньої ситуації для підтримки персоніфікованої адаптації процесу підготовки водіїв	55
2.2 Моделювання процесу обробки біометричних даних у програмних системах навчання водіїв в автошколах.....	63
2.3 Модель гібридної програмної архітектури з периферійними обчисленнями для персоніфікованої адаптації процесу підготовки в автошколі	71
Висновки до розділу 2.	82
3 МЕТОДИ ПОБУДОВИ ПЕРСОНІФІКОВАНОГО АДАПТИВНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ НАВЧАННЯ ВОДІЇВ	84
3.1 Розробка методу формування вектору профілю користувача навчальної програмної системи	84
3.2 Розробка методу побудови вектору поточного стану користувача в навчальній програмній системі	97

3.3 Розробка методу персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача	107
3.4 Розробка методу верифікації процесу вибору навчальних модулів з використанням технології інтелектуального аналізу процесів	119
Висновки до розділу 3.	129
4 ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА МОДЕЛІ ТА МЕТОДІВ ПЕРСОНІФІКОВАНОЇ АДАПТАЦІЇ ПРОЦЕСУ ПІДГОТОВКИ ВОДІЇВ В НАВЧАЛЬНИХ ПРОГРАМНИХ СИСТЕМАХ АВТОШКІЛ	131
4.1 Реалізація та експериментальна перевірка гібридної архітектури навчальної програмної системи автошколи	131
4.2 Експериментальна перевірка методів формування векторів профілю та стану користувача	139
4.3 Експериментальна перевірка методів персоніфікованої адаптації та верифікації процесу вибору навчальних модулів	148
Висновки до розділу 4.	156
ВИСНОВКИ.....	158
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ.....	161
ДОДАТОК А ПЕРЕЛІК ПУБЛІКАЦІЙ ЗДОБУВАЧА	181
ДОДАТОК Б ДОВІДКА ПРО ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОГО ДОСЛІДЖЕННЯ	183

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

AAC – навчання водінню під супроводом (фр. Apprentissage Anticipé de la Conduite)

AUC – площа під ROC-кривою (англ. Area Under the ROC Curve)

IRT – сучасна теорія тестів (англ. Item Response Theory)

HR – частота серцевих скорочень (англ. Heart Rate)

HRV – варіабельність серцевого ритму (англ. Heart Rate Variability)

GDL – багатоетапна система отримання водійських прав (англ. Graduated Driver Licensing)

MSE – середньоквадратична похибка (англ. Mean Squared Error)

ORE – дорожній екзамен (англ. On-Road Exam)

PM – інтелектуальний аналіз процесів (англ. process mining)

VDT – віртуальний іспит з водіння (англ. Virtual Driving Test)

ЕЕГ – електроенцефалографія (англ. Electroencephalography)

НПС – навчальна програмна система

ФПГ – фотоплетизмографія (англ. Photoplethysmography)

ВСТУП

Актуальність теми дослідження. Сучасний розвиток програмних навчальних систем характеризується запитом на персоніфікацію навчання з урахуванням індивідуальних когнітивних характеристик користувачів з тим, щоб динамічно адаптувати навчальні сценарії відповідно до поточних можливостей цих користувачів щодо опанування навчального матеріалу. На даний момент у більшості розвинених країн, а також і в Україні, Індустрія 4.0 набула рис сукупності взаємопов'язаних екосистем, серед яких доволі значне місце займають навчальні інформаційні системи.

Вирішення даного завдання обумовлює важливість створення та удосконалення підходів до персоніфікованої адаптації процесу навчання з комплексним урахуванням не лише досягнутого рівня знань у предметній області, а й поточного рівня когнітивного навантаження користувача, який впливає на здатність останнього ефективно вивчати навчальний матеріал. Реалізація персоніфікованої адаптації при підготовці водіїв потребує розробки комплексу методів побудови векторів профілю та стану користувача, диспетчеризації навчальних модулів з урахуванням інформації із отриманих векторів, а також верифікації знань, що використовуються для динамічного вибору навчальних сценаріїв. Сукупність цих підзадач обумовлює доцільність розробки й застосування гібридних архітектурних підходів, призначених для комплексного представлення профілю та стану користувача, а також персоніфікованої адаптації процесу підготовки в програмних навчальних системах автошкіл.

Проте існуючі адаптивні системи підготовки водіїв при виборі навчальних модулів обмежуються, як правило, оцінкою поточного рівня підготовки користувачів при підборі сценаріїв для подальшого навчання і не враховують їх поточні можливості щодо ефективного вивчення цих модулів. Генеративний штучний інтелект, що останніми роками набув значного поширення, за своєю природою не в змозі бути застосований для вирішення даних задач. Розвиток

адаптивних підходів пов'язаний із частковим врахуванням загального психофізичного стану що, однак, не забезпечує персоніфікованої адаптації процесу навчання на основі гібридних архітектурних рішень.

Для забезпечення персоніфікованої адаптації процесу навчання водінню на основі одночасного врахування профілю рівня підготовки та поточного стану користувача потребують подальшого розвитку методи та моделі адаптації процесу навчання на основі гібридних архітектур програмних навчальних систем. Крім того, набуті знання особи, яка навчається також потребують адекватної оцінки.

Таким чином, розробка моделі, методів та гібридної архітектури програмної системи для персоніфікованої адаптації процесу підготовки водіїв на основі формалізованого представлення профілю знань й поточного стану користувача є актуальним науковим завданням, вирішення якого забезпечує підвищення ефективності навчання в автошколах.

Науковим завданням дисертаційного дослідження є розробка моделі та методів персоніфікованої адаптації в програмних системах навчання в автошколі для підвищення ефективності підготовки водіїв з урахуванням як рівня підготовки користувача, так і його поточного когнітивного стану.

Зв'язок роботи з науковими програмами, планами, темами.

Робота виконана в рамках науково-дослідної теми ДР№0124U001426 «Технологія розроблення програмного забезпечення для інноваційних систем спеціального та подвійного призначення», терміни виконання: 2024 – 2026 рр.

Мета дослідження. Метою дисертаційного дослідження є розробка моделі та методів персоніфікованої адаптації процесу підготовки водіїв на базі гібридної архітектури програмної системи з використанням формалізованого представлення профілю знань та поточного стану користувача для підвищення ефективності навчання в програмних системах автошкіл.

В дисертаційному дослідженні вирішуються наступні завдання:

1. Аналіз стану проблеми адаптації та персоніфікації процесу навчання водіїв у автошколі.

2. Розробка моделі гібридної програмної архітектури з периферійними обчисленнями для персоніфікованої адаптації процесу підготовки в автошколі.

3. Розробка методів побудови вектору профілю користувача та вектору поточного стану користувача навчальної програмної системи.

4. Розробка методу персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача.

5. Розробка методу верифікації процесу вибору навчальних модулів з використанням технології інтелектуального аналізу процесів.

6. Реалізація гібридної програмної архітектури та експериментальна перевірка розроблених методів персоніфікованої адаптації процесу навчання.

Об'єктом дослідження є процес персоніфікованої адаптації в програмних системах навчання в автошколі.

Предметом дослідження є модель та методи персоніфікованої адаптації в програмних системах навчання, що враховують рівень підготовки та поточний стан користувача.

Методи дослідження: В дисертації використано методи комп'ютеризованого адаптивного тестування, машинного навчання, підтримки прийняття рішень, інтелектуального аналізу процесів та архітектурного проектування гібридних програмних систем. Метод формування вектору профілю користувача навчальної програмної системи використовує методи комп'ютеризованого адаптивного тестування та моделі адаптивних навчальних систем. Метод побудови вектору поточного стану користувача в навчальній програмній системі використовує моделі машинного навчання для оцінки варіабельності серцевого ритму. Метод персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача використовує методи підтримки прийняття рішень та теорію контекстно-адаптивних програмних систем. Метод верифікації процесу вибору навчальних модулів ґрунтується на методах інтелектуального аналізу процесів. Модель програмної архітектури з периферійними обчисленнями для персоніфікованої

адаптації процесу підготовки в автошколі розроблено із використанням гібридних програмних архітектур.

Наукова новизна одержаних результатів.

1. **Вперше** запропоновано модель гібридної програмної архітектури з периферійними обчисленнями системи навчання в автошколі, яка містить клієнтський рівень для візуалізації дорожнього середовища, периферійний рівень для диспетчеризації персоніфікованого вибору навчальних завдань та мікросервісний рівень для верифікації вибору навчальних завдань, що створює умови для персоніфікованої адаптації процесу навчання в автошколі.

2. **Удосконалено** методи побудови векторів профілю та поточного стану користувача навчальної програмної системи, які на відміну від існуючих поєднують опис як рівня знань користувача з використанням психометричних підходів, так і його поточного психоемоційного стану на основі оцінки варіабельності серцевого ритму, що створює умови для автоматизованої диспетчеризації наступних навчальних модулів з урахуванням не лише рівня підготовки користувача, а й рівня його поточного когнітивного навантаження.

3. **Вперше** запропоновано метод персоніфікованої адаптації процесу навчання шляхом вибору навчальних модулів на основі векторів профілю та стану користувача та з урахуванням ступеню новизни завдань та попередньої фільтрації модулів за рівнем складності, що дає можливість у автоматизованому режимі підібрати навчальний сценарій згідно рівня підготовки та поточного когнітивного навантаження користувача навчальної програмної системи.

4. **Дістав подальшого розвитку** метод верифікації процесу підготовки водіїв в навчальних програмних системах автошкіл, який на відміну від існуючих містить етапи побудови моделей цільового та фактичного процесів персоніфікованої адаптації навчання та виявлення порушень адаптивної логіки на основі порівняння цільової та фактичної моделей, що забезпечує умови для удосконалення правил персоніфікованої адаптації у процесі навчання з урахуванням рівня латентних знань користувачів.

Практичне значення отриманих теоретичних результатів. Практичне значення отриманих нових теоретичних результатів полягає у розробці гібридної програмної архітектури та модулів підтримки персоніфікованої адаптації навчального процесу з урахуванням рівня знань та поточного стану користувача програмної системи навчання в автошколах, що створює умови для підвищення ефективності навчання не лише з урахуванням складності опанованих користувачем навчальних сценаріїв, а й з урахуванням його поточного психоемоційного стану в умовах виникнення когнітивного перенавантаження під час навчання.

Особистий внесок здобувача. Автор отримав нові наукові результати дисертаційного дослідження самостійно. Роботи [1 – 3, 5 – 10] виконані без співавторів. У роботі [4], що була опублікована зі співавтором, представлено новий науковий результат – метод верифікації адаптивної програмної системи з використанням технології process mining.

Апробація матеріалів дисертації. Отримані в дисертаційному дослідженні результати доповідались на міжнародних науково-технічних конференціях «Проблеми інформатизації», «Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління», міжнародній науково-практичній конференції «Інтегровані інтелектуальні робототехнічні комплекси (ІРТК 2026)».

Публікації. Отримані в дисертаційному дослідженні результати опубліковано в 10 наукових працях, в тому числі 5 статей у періодичних фахових наукових виданнях України з технічних наук категорії «Б», 5 тез доповідей на міжнародних конференціях.

Структура та обсяг дисертаційної роботи. Дисертація містить вступ, чотири розділи, висновки, перелік джерел посилання із 176 найменувань, представлених на 20 сторінках, 1 додаток на 2 сторінках. Загальний обсяг дисертації складає 181 сторінку, в тому числі 158 сторінок основного тексту, анотації на 10 сторінках, 41 рисуноків, 15 таблиць, з них 3 розміщені на 3 окремих сторінках.

1 АНАЛІЗ СТАНУ ПРОБЛЕМИ АДАПТАЦІЇ ТА ПЕРСОНІФІКАЦІЇ ПРОЦЕСУ НАВЧАННЯ ВОДІЇВ У АВТОШКОЛІ

1.1 Аналіз особливостей адаптивного процесу навчання водіїв в автошколах

Процес навчання в водіїв автошколах направлений на формування таких професійних навичок та компетентностей майбутніх водіїв, що забезпечать безпечне водіння на дорогах загального користування.

Даний процес має такі основні характеристики. По-перше, до навчання допускаються курсанти з різним рівнем базової підготовки, мотивації та когнітивних здібностей. По друге, безпосередньо процес навчання включає теоретичну підготовку, практичну підготовку та контроль знань. По-третє, за результатами такої підготовки водій повинен засвоїти набір компетенцій для безпечного керування транспортним засобом [11, 12]. Характеристики основних етапів процесу навчання водіїв в автошколі наведено в табл. 1.1.

Таблиця 1.1 – Етапи процесу навчання водіїв

Етап	Характеристика	Орієнтовні витрати часу	Спосіб оцінки
1. Оцінка рівня підготовки курсанта	Визначення рівня базових знань та навичок, когнітивних здібностей, стану здоров'я	1-2 дні	Медичне обстеження, анкетування
2. Теоретична підготовка	Вивчення правил дорожнього руху та основ безпечного водіння	64 години	Комп'ютерне тестування
3. Практична підготовка	Формування психомоторних навичок водіння автомобіля	40 годин	Практичні вправи з водіння
4. Контроль знань та навичок	Складання теоретичного та практичного іспитів	2-3 години	Внутрішній та державний іспити

Процес навчання в українській автошколі характеризується суттєво більшим обсягом аудиторних годин у порівнянні з європейськими та американськими підходами, які враховують пріоритети щодо дорожньої безпеки, а також відображають національні педагогічні традиції [13 – 15]. Зазвичай, у навчальних закладах такого типу, в якості методичної основи функціонування навчальних програмних систем використовується агентне імітаційне моделювання.

Європейський підхід до навчання водіїв характеризується зменшенням кількості годин формальної підготовки та посиленою увагою до якості інструктажу.

Зокрема, французька система підготовки побудована відповідно до концепції *Apprentissage Anticipé de la Conduite* (AAC). Дана концепція передбачає двоетапну практичну підготовку [15]. Перший етап включає 20 годин теоретичного та практичного навчання в автошколі. Другий етап є етапом супроводжуваного водіння, що передбачає не менше ніж 3000 кілометрів практики управління автомобілем під наглядом досвідченого водія. Останній повинен щонайменше 5 років мати водійське посвідчення категорії В. Такий підхід дає можливість розпочинати навчання з 15-річного віку. Підготовка водіїв на етапі супроводжуваного водіння передбачає три обов'язкові зустрічі в автошколі: попередню, проміжну, що відбувається через 4 – 6 місяців навчання, та заключну зустріч, яка проводиться після проїзду 3000 км [16, 17]. Такий двоетапний підхід до підготовки водіїв на практиці забезпечує рівень успішності складання практичного іспиту близько 70% у порівнянні з рівнем успішності в 54% при традиційному навчанні в минулі роки [18].

Британський підхід передбачає обов'язковий тест сприйняття небезпеки при водінні як частину теоретичного іспиту [19]. Традиційно цей тест складається з 14 відеокліпів по одній хвилині, кожен з яких представляє дві потенційні небезпеки. При оцінюванні за кожну ідентифіковану небезпеку надається максимум 5 балів в залежності від швидкості реакції майбутнього водія. Прохідний бал становить 44 з 75 можливих. Останні 10 років при

проведенні тесту використовуються не відеокліпи, а технології комп'ютерної графіки з тим, щоб забезпечити стандартизацію оцінювання. В результаті застосування розглянутого підходу загальний показник успішності теоретичного іспиту становить 55,7% за 2021 рік [20].

Шведська програма підготовки водіїв містить компоненти ризикованого водіння Risk 1 та Risk 2 [21]. Компонент Risk 1 передбачає теоретичну підготовку водіїв з урахуванням ризиків, пов'язаних з алкоголем та втомою. Компонент Risk 2 (Riskutbildning del 2) включає тренування на треку ковзання. Даний компонент включає 3-4 години навчання щодо керування авто в критичних ситуаціях. Ці ситуації моделюються на спеціальному треку з пониженим коефіцієнтом зчеплення, або ж за допомогою системи симулювання низького зчеплення на асфальті Skid-Car [22].

Північноамериканський підхід також поєднує коротку формальну підготовку, яка включає 30-32 години теорії та 6-8 годин з інструктором, з водінням під наглядом батьків, яке триває 50-60 годин [23]. Такий процес навчання забезпечує зниження аварійності серед водіїв-початківців [24]. Зокрема, в штаті Каліфорнія використовується концепція GDL (Graduated Driver Licensing), яка використовує трьохетапний підхід до отримання водійських прав [23]. Початковий етап має тривалість 6 місяців і може починатись з 15,5 років. Проміжний етап починається з 16 років та має обмеження на водіння вночі та на кількість пасажирів. Повна ліцензія водіям може бути отримана лише через 12 місяців після початку навчання, або ж при досягненні 18-річчю. Практична підготовка включає 6 годин навчання з сертифікованим інструктором, а також 50 годин практики сімейного водіння. З них 10 годин практики мають проводитись в нічний час доби. Система GDL призначена для зниження аварійності серед підлітків.

Таким чином, міжнародна практика навчання водіїв має три ключові характеристики:

- поетапне адаптивне навчання з поступовим зняттям обмежень щодо швидкості, кількості пасажирів, водіння у темний час доби тощо, що забезпечує адаптивну підготовку водіїв у відповідності до їх поточних навичок [25];

- інтеграція у програму підготовки безпекових модулів, що призначені для навчання водінню без ризиків в небезпечних дорожніх ситуаціях та умовах, при втомі водія тощо [11];

- використання практики сімейного водіння, що забезпечує персоніфікацію практичної водійської підготовки [26].

Такі особливості міжнародної практики підготовки до водіння направлені на зменшення ризиків для водіїв-початківців на основі поступового набуття індивідуального досвіду в контрольованих умовах у відповідності до поточного рівня знань та навичок. Використання такого підходу забезпечує зниження аварій серед водіїв-початківців, зокрема віком 16 –17 років [25, 26].

На відміну від міжнародних практик, навчання в українських автошколах відрізняється великою кількістю аудиторних годин. Згідно з типовою навчальною програмою підготовки водіїв в Україні, обов'язковими є 64 академічні години теоретичної підготовки та 40 годин практичного навчання [11, 12]. Проте, на відміну від європейського та американського підходів, при вітчизняній підготовці не передбачається обов'язкове набуття практики водіння під наглядом за межами автошколи [27, 28]. Дане обмеження збільшує строки формування стійких навичок водіння та адаптації до небезпек та ризиків, що виникають у реальних дорожніх умовах.

Таким чином, ефективність підготовки водіїв значною мірою залежить від якісної інтеграції отриманих теоретичних знань і практичних навичок, що потребує поетапного засвоєння теорії та практики водіння [29]. Традиційний паралельний підхід до навчання водіїв в автошколах, який передбачає одночасне вивчення теорії та практики, створює підвищене когнітивне навантаження внаслідок надмірних вимог до запам'ятовування [30, 31]. Тому для забезпечення ефективної інтеграції знань та навичок водіння з 2024 року в нашій країні запроваджено послідовний підхід, коли доступ до практичного навчання

надається майбутньому водієві лише після успішного складання теоретичного іспиту [22, 23]. Такий підхід базується на когнітивних теоріях навчання [34, 35] які визначають необхідність формування концептуальних знань як основи для набуття процедурних навичок. Таким чином, послідовний підхід дає можливість курсантам автошколи на базовому рівні сформувати стійкі знання щодо правил дорожнього руху, а потім інтегрувати їх з практичними навичками з водіння автомобілем.

Наведені в табл. 1.1 етапи процесу навчання водіїв в автошколі мають наступні особливості.

Етап медичного обстеження і первинної діагностики призначений для оцінки придатності кандидатів до навчання водіння [36], для цього етапу, зокрема, характерне застосування спеціалізованих діагностичних моделей. Медична діагностика зазвичай включає офтальмологічне, наркологічне, неврологічне, терапевтичне, а також психіатричне обстеження. Мета діагностики полягає у виявленні протипоказань щодо керування транспортними засобами [37].

Теоретична підготовка орієнтована на формування у курсанта автошколи системи знань щодо безпечного водіння в різних дорожніх ситуаціях [38]. З 2024 року майбутні водії мають вибір між навчанням в автошколі, яке займає 64 аудиторних години, та самостійною теоретичною підготовкою [32]. Однак на практиці самостійна теоретична підготовка вимагає високого рівня самоорганізації від майбутнього водія та може призводити до формування неповних знань щодо правил дорожнього руху [39].

Етап теоретичної підготовки включає виконання наступних модулів [11, 32]:

- вивчення правил дорожнього руху;
- засвоєння основ безпеки дорожнього руху;
- вивчення побудови автомобіля, а також правил його експлуатації;
- засвоєння медичного забезпечення;
- вивчення особливостей електромобілів;

- вивчення психологічних аспектів водіння.

Етап практичного навчання складається з трьох кроків [40]:

- формування базових навичок управління автомобілем на автомайданчику;

- водіння в простих дорожніх умовах;

- адаптація до міського середовища з інтенсивним рухом.

Крок формування базових навичок управління автомобілем передбачає засвоєння психомоторних навичок водіння в контрольованому середовищі на автомайданчику [41]. Майбутні водії опановують навички роботи з педалями, кермовим колесом та важелями управління. На даному кроці мають бути сформовані не лише відчуття габаритів автомобіля, а й правильні рухові навички [42]. Типові вправи даного кроку включають: рух з місця, зупинку автомобіля, повороти на місці, паралельне паркування авто, а також заїзд у гараж та розворот у обмеженому просторі. Всі ці вправи мають бути засвоєні на рівні автоматичного виконання [43].

Крок водіння в простих дорожніх умовах виконується на вулицях з мінімальним дорожнім трафіком. Основним завданням даного кроку є інтеграція в умовах реального дорожнього середовища отриманих на попередньому етапі теоретичних знань з практичними навичками водіння. Курсанти мають навчитись утримувати поточну смугу руху, дистанцію до попереднього автомобіля, а також дотримуватись швидкісного режиму [44].

Крок адаптації до міського середовища з інтенсивним рухом передбачає водіння в умовах інтенсивного міського трафіку, що потребує розпізнавання потенційної небезпеки, пов'язаної, зокрема, з неуважними пішоходами, раптовими змінами смуг руху, агресивним водінням інших водіїв, а також несприятливими погодними умовами [45]. Курсанти відпрацьовують складні маневри: проїзд перехресть, перестроювання, обгони, паркування в реальних умовах. Особлива увага приділяється нічному водінню як найбільш складному аспекту практичної підготовки.

На етапі практичного навчання формуються психомоторні навички з водіння. Процес формування навичок включає ряд стадій. Спочатку курсант має свідомо контролювати кожний рух [40]. Друга стадія характеризується поступовою автоматизацією базових дій. На третій стадії курсант володіє навичками автономного водіння без свідомого контролю [41]. На практиці автоматизація базових навичок водіння відбувається після 50 годин активних тренувань [46]. До цього моменту час реакції водія поступово зменшується до приблизно 0,7 секунд на очікувані ситуації та 1,5 секунди на несподівані загрози [47 – 49].

Таким чином, наведені три кроки призначені для формування функцій безпечного водіння в різних дорожніх умовах. В [50] показано, що такі функції належать до класу виконавчих функцій. Успішність виконавчих функцій в першу чергу базується на роботі оперативної пам'яті людини, тобто здатності тримати в фокусі уваги визначену кількість завдань, а також гальмуванні неактуальних імпульсивних реакцій (наприклад, таких як різке гальмування, що не відповідає дорожній обстановці або ж, навпаки, ігнорування поточної небезпеки).

В цілому когнітивний процес засвоєння теоретичних та практичних знань при навчанні водіїв включає сприйняття, перетворення, зберігання та використання знань щодо водіння в різних дорожніх умовах. Об'єм оперативної (робочої) пам'яті людини суттєво впливає на якість прийняття рішень у складних дорожніх умовах, в тому числі на швидкість оновлення важливої (пріоритетної) інформації, а також на побудову алгоритму дій з подолання потенційної небезпеки. Когнітивна гнучкість, зокрема представлена як вміння швидко перемикається між завданнями при водінні дає можливість суттєво зменшити ризик критичних помилок при зміні дорожньої ситуації.

Згідно теорії когнітивного навантаження [51], оперативна пам'ять людини має суттєві обмеження. Людина здатна запам'ятовувати 7 ± 2 [52] елементи та ефективно обробляти лише 4 ± 1 інформаційних елементів [53, 54] одночасно. Типовий курс із засвоєння правил дорожнього руху передбачає вивчення більш ніж 200 дорожніх знаків, а також велику кількість правил визначення пріоритету

для дорожнього руху та численні винятки. Такий об'єм інформації створює надмірне когнітивне навантаження, що суттєво перешкоджає навчанню [55, 56].

Аналіз особливостей процесу навчання в автошколі показав, що основні проблеми когнітивного перевантаження при засвоєнні теоретичного матеріалу включають:

- інформаційне перевантаження, яке зазвичай виникає при одночасному вивченні декількох категорій дорожніх знаків та приводить до формування плутанини у знаннях курсанта автошколи [57].

- відсутність або неповнота контексту використання знань, що виникає внаслідок абстрактного вивчення правил дорожнього руху без їх прив'язки до відповідних дорожніх ситуацій та ускладнює засвоєння цих правил [58].

- відсутність системного підходу внаслідок фрагментарного подання навчального матеріалу без відображення системних зв'язків між правилами дорожнього руху [59].

Відповідно, у людей з різним рівнем когнітивних можливостей спостерігається збільшення/зменшення рівня аварійності, різна інтерпретація однотипних дорожніх ситуацій тощо внаслідок перевантаження пам'яті майбутніх водіїв при одночасному засвоєнні великих обсягів різномірної інформації [60].

Типові помилки при теоретичному навчанні включають в себе:

- некоректне використання подібних понять, наприклад зупинка або стоянка, обгін або випередження внаслідок інформаційного перевантаження [60, 61];

- використання знань без урахування контексту: правила дорожнього руху, засвоєні в одному контексті, використовуються в непридатних ситуаціях [62, 63];

- протиріччя між знаннями, що виникають внаслідок відсутності системного підходу при викладанні; раніше засвоєні помилкові залежності заважають формуванню коректних знань, або ж нова інформація спотворює раніше засвоєні знання [64, 65].

Проведений аналіз показав, що при практичній підготовці водіїв помилки зазвичай виникають при виконанні паралельного паркування внаслідок неточної траєкторії руху, при виконанні лівого повороту внаслідок неправильної оцінки дистанції, при нічному водінні внаслідок порушення сприйняття відстаней курсантом автошколи, а також при водінні в екстремальних умовах внаслідок втрати контролю над автомобілем.

Вказані помилки викликані наступними ключовими причинами:

- недостатні навички просторової орієнтації;
- недостатня координація між сприйнятим зором зображення поточної дорожньої обстановки та моторними реакціями водія;
- відсутність розподілу уваги між кількома завданнями з водіння одночасно.

Ці помилки є наслідками переважного використання пасивних методів навчання замість інтерактивних, недостатнього використання візуальних засобів при навчанні; слабкого зворотного зв'язку щодо поточного рівня знань та когнітивного навантаження курсанта автошколи.

В результаті у останні роки тільки 21% кандидатів у водії складають теоретичний іспит та лише 16% кандидатів складають практичний іспит з першої спроби [66]. Для порівняння, у Великобританії теоретичний та практичний тест з першої спроби складають 44 – 46% та 46 – 49% кандидатів [67].

Виконаний аналіз міжнародного досвіду підготовки водіїв дає можливість зробити висновок, що розглянуті проблеми навчання курсантів в автошколі є наслідком розриву між теоретичною та практичною підготовкою. Цей розрив має наступні особливості:

- абстрактний характер теоретичного навчання із слабкою прив'язкою до конкретних ситуацій;
- обмежений час практичних занять щодо прийняття рішень в умовах обмеженого часу;
- недостатньо уваги приділяється тренуванню навичок швидкого аналізу дорожньої обстановки;

– слабка інтеграція правил дорожнього руху з технікою управління автомобілем.

Для вирішення наведеної проблеми розриву теоретичного та практичного навчання потрібно, з урахуванням міжнародного досвіду, виконати персоніфіковану адаптацію навчальних завдань. Така адаптація орієнтована на поступальний розвиток когнітивних навичок курсанта автошколи.

Використання персоніфікованої адаптації процесу підготовки має забезпечити наступні можливості навчання водіїв в автошколі:

– багаторазове повторення складних сценаріїв управління автомобілем з урахуванням рівня підготовки та поточного стану (стрес, спокійний стан) курсанта автошколи;

– створення персоніфікованих умов навчання без впливу зовнішніх факторів;

– адаптація рівня складності завдань відповідно до поточного індивідуального рівня підготовки курсанта автошколи.

Сценарії навчання мають забезпечувати:

– імітацію екстремальних погодних умов (дощ, сніг, туман);

– моделювання аварійних ситуацій при управлінні автомобілем та відпрацювання реакцій на ці ситуації;

– тренування сприйняття небезпеки при водінні автомобіля;

– відпрацювання складних маневрів та складних дорожніх ситуацій з необмеженою кількістю повторів.

Задоволення наведених вимог потребує реалізації адаптивного навчання водіїв в автошколі з наступними можливостями:

– регулювання темпу навчання у відповідності до когнітивного навантаження курсанта автошколи;

– виявлення типових персональних помилок курсанта автошколи та підбір персоніфікованих завдань для засвоєння знань та навичок, що забезпечують усунення цих помилок;

- персоніфіковане коригування послідовності подачі навчальних матеріалів курсанту автошколи;
- прогнозування рівня успішності при складанні іспитів.

Слід зазначити, що реалізація персоніфікації адаптивного навчання водіїв автошколи потребує розробки відповідних програмних архітектур для програмних навчальних систем з можливостями автоматизованого збору та аналізу даних про стан когнітивного навантаження курсанта у процесі його навчання в автошколі. Зворотний зв'язок на основі цих даних дає можливість адаптувати процес навчання з урахуванням персонального рівня підготовки курсанта та його поточного стану.

1.2 Аналіз архітектур навчальних програмних систем

Архітектура програмного забезпечення інформаційних систем визначає структурні елементи ПЗ, їх взаємозв'язки та принципи організації взаємодії між цими структурними елементами. Вибрана архітектура ПЗ впливає на життєвий цикл розробки інформаційної системи. В умовах швидкого розвитку інформаційних технологій, широкого застосування інтелектуальних підходів, рішення щодо архітектури ПЗ мають забезпечувати адаптивність, а також масштабованість та надійність програмних систем, що реалізують ці технології [68].

Сучасні підходи до розробки програмних архітектур базуються на використанні архітектурних патернів. Такі патерни повинні забезпечувати процеси децентралізованого навчання та виконання відповідних завдань. Останні є перевіреними на практиці типовими системними рішеннями для визначених класів задач. Архітектурні патерни визначають такі характеристики архітектури [69]:

- загальну структуру програмної системи;
- порядок взаємодії компонентів програмної системи;
- типи і значення атрибутів системи.

Архітектура навчальної програмної системи (НПС) задає організацію компонентів системи і механізми їх взаємодії і тому визначає можливості НПС щодо реалізації персоналізованого підходу до навчання. При персоніфікації процесу навчання навчальна платформа має забезпечувати масштабованість, адаптивність і прозорість прийнятих рішень щодо вибору навчальних сценаріїв [69]. Також важливу функцію у таких платформах виконують моніторингові системи.

Згідно цих вимог виділено такі критерії класифікації архітектурних підходів:

- ступінь централізації компонентів;
- підтримка адаптації та персоніфікації процесу навчання;
- масштабованість;
- інтероперабельність.

Загальну характеристику цих критеріїв наведено в табл. 1.2.

Таблиця 1.2 – Критерії класифікації архітектурних підходів

№	Критерій	Зміст	Призначення
1	Ступінь централізації компонентів	Концентрація функцій в монолітному ядрі програмної системи / розподіл функцій між незалежними модулями	Визначає здатність архітектури програмної системи до адаптації при зміні вимог
2	Підтримка адаптації та персоніфікації процесу навчання	Вбудовані механізми побудови персоніфікованих траєкторій навчання	Визначає можливість поєднання персоніфікації та адаптації процесу навчання
3	Масштабованість	Забезпечення рівня продуктивності при збільшенні аудиторії користувачів	Є необхідною умовою практичного застосування навчальної системи
4	Інтероперабельність	Підтримка відкритих стандартів обміну даними, наприклад xAPI, SCORM, IMS LTI	Забезпечує інтеграцію в гетерогенне середовище

Критерій ступеню централізації компонентів програмної системи враховує сконцентрованість функцій системи в єдиному монолітному ядрі або ж розподіл їх між різними незалежними модулями. Даний критерій визначає можливості адаптації архітектури до нових вимог [70].

Критерій підтримки адаптивності та персоніфікації визначає наявність вбудованих механізмів формування індивідуальних навчальних траєкторій, тобто підбору та уточнення контенту під час навчання на основі інформації про поточний стан користувача [71, 72].

Критерій масштабованості визначає здатність програмної системи зберігати рівень продуктивності при збільшенні кількості підключених користувачів. Масштабованість розглядається як необхідна передумова практичного застосування, оскільки навчальна система не має деградувати при збільшенні аудиторії [73].

Критерій інтегрованості відображає можливість підтримки відкритих стандартів щодо обміну навчальними даними. Даний критерій відображає здатність навчальної системи взаємодіяти із зовнішніми системами [74, 75].

На основі зазначених критеріїв виділено чотири типи архітектур, які розвивались як архітектурні покоління. Термін «покоління» використано для відображення еволюційних змін у функціональних можливостях архітектур, тобто кожне покоління архітектур долає системні обмеження попереднього покоління.

Перше покоління включає монолітні та клієнт-серверні архітектури. Монолітна архітектура реалізує у одному модулі всі функції, наприклад автентифікацію, управління контентом, оцінювання, а також звітність. Клієнт-серверна архітектура розділяє рівні представлення даних та бізнес-логіки. Проте зазвичай обидва рівня, як бізнес-логіка, так і сховище даних централізовано розміщуються на стороні сервера.

Типовими навчальними програмними системами цього покоління є перші версії Moodle, WebCT та Blackboard [76, 77]. НПС першого покоління включали

реєстрацію навчальних курсів, тестування, підтримку журналу оцінок, а також форумів для комунікації з учнями. Такі системи використовували стандарт електронного навчання SCORM 1.2, який забезпечував перенесення навчальних курсів між різними платформами, а також стандарт AICC, попередник SCORM [78].

Архітектури даного типу мають такі обмеження для задач адаптивного навчання:

- надається однаковий контент для всіх користувачів без урахування індивідуального рівня знань і можливостей навчання, оскільки в архітектурі не передбачено моделювання учня [79];
- складне оновлення окремих функцій, оскільки зміна коду потребує повної перекомпіляції й повторного розгортання НПС, що обмежує розвиток навчальної платформи [80];
- масштабування виконується шляхом нарощення потужності сервера, що обмежує використання навчальної програмної системи при великій кількості користувачів [73].

Необхідність подолання цих обмежень привела до розробки програмних архітектур наступного покоління [80].

Другий тип програмних архітектури реалізує мікросервісну парадигму. Дана парадигма передбачає розбиття цілісної програмної системи на сукупність незалежних сервісів (або мікросервісів). Кожен із мікросервісів виконує визначену функцію системи, наприклад автентифікацію, оцінювання, управління начальними сценаріями тощо. Сервіси взаємодіють через програмні інтерфейси та шину подій. Шина подій призначена для асинхронного обміну повідомленнями між сервісами. Якщо один сервіс опублікує подію, то усі інші сервіси отримують сповіщення про дану подію і можуть незалежно обробити її. Мікросервіси можуть бути розгорнуті в хмарі в контейнерах, тобто в ізольованих програмних середовищах. Для розгортання використовуються Docker-контейнери, які оркеструється засобами системи Kubernetes [81 – 83].

Архітектура навчальних програмних систем даного типу представлена платформами Canvas LMS, Coursera, edX. Хмарна платформа Canvas LMS призначена для застосування у вищій освіті та корпоративному навчанні. Платформа підтримує створення курсів, автоматизоване оцінювання, відеоконференції, аналітику академічної успішності. Відкрита онлайн-платформа Coursera пропонує тисячі навчальних курсів і призначена для здобуття неформальної освіти. Платформа edX, заснована Гарвардським університетом і МІТ; підтримує адаптивне навчання за рахунок використання мікросервісів [84, 85].

Використання мікросервісів в цих системах дає можливість розподілити навантаження між вузлами обробки. Автоматизований процес розгортання системних оновлень забезпечує постійне доповнення функцій навчальних систем [82].

Аналіз архітектур даного типу показав, що вони мають такі обмеження при вирішенні задач адаптивного навчання:

- функції адаптації надаються як окремі сервіси без попереднього системного проєктування взаємодії цих сервісів [86];
- типова мікросервісна архітектура не використовує централізоване зберігання динамічного профілю користувача, так щоб цей профіль був доступним для всіх компонентів системи одночасно [87];
- адміністрування великої кількості мікросервісів потребує додаткових витрат на інфраструктуру автоматизованого розгортання [88].

Таким чином, архітектура даного типу забезпечує масштабованість, але має обмеження щодо підтримки розгортання та персоналізації.

Третє архітектурне покоління охоплює адаптивні й інтелектуальні підходи. Ключовою ознакою архітектур даного типу є вбудована моделі учня, яка є центральним архітектурним компонентом. Дана архітектура використана в адаптивних освітніх гіпермедійних системах [89, 90].

Архітектура освітньої гіпермедійної системи включає:

- модель предметної області, яка визначає структуроване представлення навчальних матеріалів на основі онтологій; таке представлення задає навчальні теми і залежності між ними, цілі навчання і використовується для побудови навчальних траєкторій [91];

- модель учня, що містить дані про рівень знань користувача щодо кожної теми, стиль навчання, цілі навчання, темп засвоєння матеріалу та поведінкові патерни; дана модель обумовлює можливість персоніфікації навчання [92];

- рушій адаптації як механізм генерації персоналізованої послідовності вивчення навчального контенту; рушій використовує моделі учня та предметної області і надає рекомендації щодо подальших кроків навчання, порівнюючи поточний стан користувача із його цільовим станом [93].

Архітектури даного покоління використовуються в адаптивних освітніх гіпермедійних системах WHURLE, TANGOW, IL-Web, Knewton, ASSISTments, DreamBox [94, 95].

Адаптивні архітектури мають такі обмеження для задачі персоніфікованого адаптивного навчання:

- модель учня, як правило, є статичною і тому не в повній мірі відповідає поточному стану знань користувача [92];

- точність адаптації залежить від онтології предметної галузі, проте підтримка онтології потребує суттєвих витрат часу при удосконаленні змісту дисциплін та появі нових дисциплін [96];

- адаптація виконується переважно з використанням явних правил; моделі машинного навчання для виявлення неявних поведінкових патернів не використовуються [97].

Четверте покоління архітектур використовує агентні та рекомендаційні підходи, що передбачають використання тестових моделей і тестових сценаріїв. Згідно агентному підходу програмна система складається з сукупності спеціалізованих програмних агентів. Такі агенти, наприклад педагогічний агент, агент контенту, агенти оцінювання та моніторингу, автономно оцінюють стан середовища та приймають рішення у відповідності зі своєю цільовою функцією.

Взаємодія між автономними агентами задає поведінку програмної системи. Така поведінка зазвичай орієнтована на реалізацію індивідуальної роботи з користувачем [98]. Агентна архітектура може бути основана на мікросервісах, коли кожен агент реалізується в окремому контейнері. На відміну від мікросервісів агенти мають внутрішню пам'ять і приймають рішення на основі свого поточного стану і стану середовища. Агент може змінити поведінку без повторної компіляції, а мікросервіси потребують зміни коду. Агентна архітектура задає правила взаємодії і є більш високим рівнем абстракції, ніж мікросервіси, оскільки останні задають порядок розгортання програмної системи.

Рекомендаційна архітектура у складі НПС представлена у вигляді самостійного архітектурного компоненту із визначеними інтерфейсами, вимогами до масштабованості та правилами взаємодії з іншими модулями системи. Вихід рекомендаційного компоненту змінюється в залежності від поточного стану моделі користувача та контексту навчання. Рекомендаційна архітектура складається з чотирьох шарів: збору та попередньої обробки даних; формування ознак; побудови рекомендацій; навчання.

Алгоритмічна логіка рекомендаційного компоненту базується на використанні комбінації таких підходів:

- колаборативна фільтрація, яка передбачає, що рекомендації щодо навчання формуються на основі виявлення закономірностей у поведінці статистично схожих користувачів, наприклад, «учні, які опанували теми А та В, далі переходили до вивчення теми С» [99, 100];

- фільтрація на основі змісту, коли рекомендації щодо навчальних модулів ґрунтуються на характеристиках навчальних даних з урахуванням поточного рівня знань конкретного учня;

- знанієвий підхід, коли навчальна система використовує знання у формі формальних правил та онтології предметної області для підбору наступного навчального модуля [99].

Зазначені підходи реалізують логіку побудови рекомендацій. Архітектурним рішенням четвертого покоління є сам рекомендаційний компонент із заданим місцем у топології НПС, визначеними API-контрактами та заданою схемою взаємодії з профілем учня.

Рекомендаційні та агентні механізми використовують такі навчальні системи як AutoTutor, Cognitive Tutor, Coursera [101, 102].

Інтелектуальна система підтримки навчання AutoTutor веде навчання природною мовою і адаптується в залежності від відповідей студента. Наприклад, коли користувач відповідає на питання щодо принципів роботи процесора, то AutoTutor виконує порівняння відповіді зі списком очікуваних понять з використанням латентно-семантичного аналізу, виявляє протиріччя. Потім система генерує послідовність підказок у вигляді ієрархії, спочатку найзагальніше питання «Що ви можете додати до відповіді?», потім більш детальні підказки. Система надає пряму відповідь лише у випадку, коли користувач на відповідь за допомогою підказок. Такий адаптивний підхід підвищує результати навчання порівняно із самостійним опанування підручника.

Cognitive Tutor є системою, що основана на теорії адаптивного управління мисленням [103]. За цією теорією кожен навчальний крок студента має зіставлятись з моделлю когнітивних навичок. За результатами порівняння навчальна система оцінює ймовірність оволодіння студента кожною навичкою і потім адаптивно вибирає навчальні задачі. Наприклад, якщо при опануванні курсу алгебри користувач систематично робить помилки при складанні і вирішенні рівнянь з двома змінними, навчальна система збільшує частку схожих вправ і надає покрокові повідомлення-підказки при виникненні проблем.

Рекомендаційні модулі Coursera аналізують більш ніж сто параметрів, зокрема поточний прогрес курсу, тривалість перегляду лекцій, а також результати тестів, поведінку схожих слухачів. На основі аналізу Coursera пропонує наступний навчальний курс з детальним поясненням, у якому вказує відсоток схожих користувачів, що обрали рекомендований курс. Аналогічний підхід реалізовано в LinkedIn Learning [104, 105].

Агентний підхід забезпечує адаптацію рекомендацій щодо навчання навіть для нових непередбачуваних ситуацій і дає можливість враховувати як поточний стан знань користувача, так і цілі його навчання.

Проте агентні та рекомендаційні архітектури мають наступні обмеження:

- рекомендаційний модуль і головний модуль навчальної програмної системи функціонують окремо, з обмеженим обміном даними;
- не використовується єдиний протокол передачі моделей користувачів різними агентами [98];
- рекомендаційний підхід потребує великих обсягів навчальних даних, на малих обсягах система навчання має низьку точність підбору навчальних матеріалів;
- проблема «холодного старту» виникає для нових щойно запущених курсів, для яких відсутні навчальні дані [96];
- агентні системи не завжди мають механізми пояснення прийнятих рішень, що обмежує контроль викладачами траєкторії навчання [97].

Важливою особливістю є недоцільність використання при побудові таких архітектур традиційних методів оптимізації.

Результати порівняльного аналізу розглянутих архітектур наведено у табл. 1.3.

Таблиця 1.3 – Результати порівняльного аналізу програмних архітектур

Властивості	Монолітна / Клієнт- серверна	Мікросервісна	Адаптивна	Агентна / рекомендаційна
Адаптивність	–	±	+	+++
Масштабованість	±	+++	±	±
Персоніфікація (вбудована модель учня)	–	–	+	+
Інтерпретованість рішень	+	+	±	–
Інтероперабельність	±	+	±	±
Складність впровадження	низька	висока	висока	дуже висока

В табл. 1.3 використано такі умовні позначення:

- +++ – повна підтримка;
- + – підтримка;
- ± – обмежена підтримка;
- – відсутність підтримки.

Результати порівняльного аналізу архітектур дозволяють зробити висновок, що жоден з розглянутих архітектурних підходів не має набору властивостей адаптивності, масштабованості, персоніфікації, інтерпретованості та інтероперабельності, необхідних для персоніфікованої адаптації процесу навчання.

Адаптивні та агентні системи мають обмежену масштабованість та характеризуються непрозорістю рішень. Мікросервісна архітектура забезпечує масштабованість, проте має обмеження щодо персоніфікації та механізму адаптації. Монолітна архітектура не підтримує адаптивність і персоніфікацію і має обмежену масштабованість.

При переході від монолітної до мікросервісної архітектури вирішуються обмеження щодо масштабованості та інтероперабельності, проте залишається обмеження персоніфікації та адаптивності. При переході до адаптивної архітектури вирішуються проблеми адаптивності та персоніфікації проте виникають проблеми масштабованості, інтерпретованості та інтероперабельності. При переході до агентної архітектури виникає обмеження інтерпретованості [94, 99].

Таким чином, на основі аналізу архітектур навчальних програмних систем виявлено ряд обмежень цих архітектурних підходів щодо задачі реалізації персоніфікованого адаптивного навчання у масштабованих системах. Характеристики цих обмежень наведено в табл. 1.4.

Архітектурне протиріччя між адаптивністю та масштабованістю полягає в тому, що масштабовані архітектури розробляються в першу чергу з урахуванням ефективного розподілу навантаження, а адаптивні модулі інтегруються такі навчальні програмні системи у вигляді зовнішніх сервісів. Масштабована

платформа та механізми персоніфікації функціонують в даній ситуації паралельно і не представляють як єдину архітектурну систем. Тобто ці складові не були спроектовані для скоординованої взаємодії [83, 86].

Відсутність актуальної централізованої моделі користувача пов'язана з тим, що адаптивні навчальні системи використовують статичні профілі учнів або децентралізоване представлення при використанні агентних архітектур. Такі профілі не відображають актуальний стан рівня підготовки користувача.

Відсутність єдиного профілю користувача, що оперативно оновлюється та є доступним для всіх модулів, унеможливорює синхронну адаптацію на рівні навчальної системи в цілому, тому що окремі модулі отримують різну інформацію про стан користувача і приймають неузгоджені рішення на основі неузгоджених або неактуальних даних [92, 93].

Таблиця 1.4 – Обмеження архітектур в існуючих навчальних програмних системах

Обмеження	Опис обмеження	Вплив на реалізацію персоніфікованої адаптації процесу навчання	Напрямки усунення недоліку
Протиріччя між можливостями адаптивності та масштабованості	Масштабовані архітектури розробляються з урахуванням ефективного розподілу навантаження, а адаптивні модулі інтегруються в такі навчальні програмні системи у вигляді зовнішніх сервісів	Ускладнена взаємодія між модулями персоніфікації та базовою навчальною платформою	Розробка гібридної архітектури, в якій передбачено і масштабованість і адаптивність в єдиній системі

Закінчення таблиці 1.4

Обмеження	Опис обмеження	Вплив на реалізацію персоніфікованої адаптації процесу навчання	Напрямки усунення недоліку
Відсутність актуальної централізованої моделі користувача	Профіль учня статичний або розподілений між агентами і не відображає поточний стан користувача	Різні модулі використовують неузгоджені дані, що обмежує можливості персоніфікації	Розробка єдиної моделі користувача (учня), що відображає його поточний стан і використовується всіма компонентами навчальної програмної системи
Обмежена інтерпретованість процесу навчання та можливостей адаптації даного процесу	Рекомендаційні та агентні модулі не містять механізмів інтерпретації рішень.	Викладач не може перевірити правильність адаптації траєкторій навчання	Розробка інтерпретованого підходу до персоніфікованої адаптації процесу навчання

Недостатня інтерпретованість процесу адаптації виникає тому, що адаптивні рекомендаційні та агентні модулі не мають механізмів пояснення прийнятих рішень [107, 108]. Дане обмеження не дає змоги контролювати траєкторії навчання і знижує рівень довіри викладачів та інструкторів [109].

Таким чином, персоніфікована адаптація процесу навчання потребує розроблення гібридної архітектури, що поєднує переваги масштабованості мікросервісної архітектури, можливості динамічної моделі користувача та інтерпретованості монолітної архітектури.

1.3 Дослідження підходів до персоніфікації навчання

Аналіз програмних архітектур у підрозділі 1.2, засвідчив, що існуючі архітектурні рішення не враховують одночасно рівень знань користувача та його

поточний стан як два різних виміри процесу навчання, які взаємно доповнюють інформацію про користувача.

Адаптація процесу навчання ґрунтується на інформації про рівень знань користувача. Навчальна програмна система визначає, які навчальні модулі і які знання вже засвоєно, і підбирає навчальний матеріал схожої складності [110, 111]. Персоніфікація, натомість, базується на поточному когнітивному стані учня. Навчальна система визначає поточний рівень когнітивного навантаження користувача і формат подання та зміст навчального матеріалу з метою підтримувати ефективний рівень сприйняття навчального матеріалу користувачем [112]. Адаптація дає відповідь на питання «Який навчальний матеріал, якого рівня складності надати користувачеві?», а персоніфікація – «Як і який подавати матеріал в поточний момент, щоб навчання було ефективним, без перевантаження?». Адаптивна навчальна система без персоніфікації підбирає модуль відповідної до рівня знань користувача складності, проте в момент когнітивного перенавантаження учень не зможе засвоїти матеріал.

Даний підрозділ присвячений аналізу підходів до персоніфікації в навчальних системах, а також виявленню вимог до персоніфікації у контексті навчання водінню.

Персоніфікація враховує когнітивне навантаження, яке відображає ступінь розумових зусиль людини, що мають бути використані при виконанні навчального завдання. Когнітивне навантаження пов'язано із обробкою інформації, що розміщена в робочій пам'яті. Теорія когнітивного навантаження [113] була розроблена в 1980-х роках на базі досліджень в області когнітивної психології та досліджень щодо архітектури людського розуму. Мета створення цієї теорії полягала в тому, щоб пояснити вплив когнітивних обмежень людини на ефективність навчання, а також на формування відповідних навичок.

Дана теорія враховує відмінності між робочою та довготривалою пам'яттю людини. Робоча пам'ять призначена для тимчасового зберігання та оперативної обробки інформації. В робочій пам'яті одночасно може бути розміщена

обмежена кількість елементів, в більшості випадків від 4 [114] до 7 [115] з обмеженим часом зберігання, зазвичай до 30 секунд. Довготривала пам'ять має високу ємність й призначена для тривалого зберігання знань та навичок. На відміну від робочої, довготривала пам'ять людини зберігає представлені у вигляді схем структури знань. Схеми дають можливість об'єднати набір інформаційних елементів в один блок, що приводить до зменшення навантаження на пам'ять. На практиці місткість довготривалої пам'яті вважається необмеженою. Обидва типи пам'яті знаходяться у постійній взаємодії. Робоча пам'ять призначена для поточної обробки та кодування інформації. Довготривала пам'ять зберігає та дає можливість відтворити отримані знання та процедури їх обробки. Перевищення ємності робочої пам'яті призупиняє процес формування нових когнітивних схем і унеможливорює ефективне засвоєння навчального матеріалу [113]. Згідно когнітивного навантаження, навчальні матеріали мають бути сформовані таким чином, щоб мінімізувати або зменшити зайві вимоги до робочої пам'яті людини.

В дослідженнях [113, 116] виділено внутрішнє, зовнішнє та релевантне навантаження. Внутрішнє навантаження залежить від складності навчального матеріалу, від кількості елементів у матеріалі і зав'язків між ними. При вивченні окремих фактів спостерігається мінімальне навантаження, при розв'язанні комплексної задачі із множиною умов когнітивне навантаження буде найбільшим. Стороннє навантаження залежить від змісту матеріалу. Навантаження збільшується, якщо навчальний матеріал містить суперечливу інформацію, представлено з використанням незручного інтерфейсу тощо. Стороннє навантаження потрібно зменшувати для підвищення ефективності навчання. Релевантне навантаження виникає при формуванні нових когнітивних схем. Таке навантаження доцільно збільшувати. Так, якщо користувач навчальної програмної системи в автошколі декілька разів повторює алгоритм виконання лівого повороту і досягає автоматизованого виконання повороту, то в даному випадку використовується корисне навантаження. Це місце в робочій

пам'яті для подальшого аналізу ситуації на дорозі [117]. Однак сума всіх трьох навантажень має бути не більше, ніж можливості робочої пам'яті.

При персоніфікації навчальна система має утримувати користувача у зоні оптимального потоку [118], тобто в межах робочої пам'яті з максимальним значенням релевантного навантаження.

Когнітивний стан змінюється достатньо швидко, упродовж секунд або хвилин в залежності від складності навчального завдання, рівнів стресу та втоми, що відрізняє його від повільно зростаючого рівня знань. Ця відмінність обумовлює необхідність безперервного вимірювання поточного когнітивного стану користувача [112].

Персоніфікація на основі поточного когнітивного навантаження потребує його постійного вимірювання і передачі цієї інформації у навчальну програмну систему. Використовують чотири групи методів вимірювання когнітивного навантаження: фізіологічні; поведінкові; суб'єктивні, на основі продуктивності [119].

Характеристики цих методів наведено у табл. 1.5.

Фізіологічні методи забезпечують найвищу точність оцінювання когнітивного навантаження. Електроенцефалографія (ЕЕГ) фіксує зміни потужності тета і альфа-ритмів як маркерів когнітивного навантаження [120 – 121]. Варіабельність серцевого ритму може бути виміряна носимими пристроями [122].

Таблиця 1.5 – Методи вимірювання когнітивного навантаження

Група методів	Перевага	Обмеження	Можливості використання у НПС автошколи
Фізіологічні, які реєструють біометричну інформацію, зокрема мозкову активність, серцевий ритм, реакцію зіниці як прямі індикатори стану користувача	Об'єктивна оцінка когнітивного навантаження у реальному часі завдяки прямій реєстрації біометричних сигналів	Висока інвазивність і вимоги до спеціалізованого обладнання суттєво обмежують масштабованість і впровадження у стандартних навчальних середовищах	Точний сенсорний сигнал для виявлення піків навантаження і калібрування персоніфікованих моделей
Поведінкові, аналізують дії учня під час роботи з НПС, зокрема затримки, рухи, патерни взаємодії як ознаки навантаження	Повна відсутність інвазивності та інтеграція у НПС без додаткового обладнання	Нижча точність порівняно з фізіологічними методами; сигнали можуть бути спотворені зовнішніми чинниками	Основний пасивний фоновий моніторинг когнітивного навантаження упродовж усієї навчальної сесії без втручання у навчальний процес
Суб'єктивні, отримують самооцінку учня щодо навантаження через опитувальники після завершення завдання	Забезпечує інтерпретовані дані з позиції самого учня	Виключно ретроспективні, вимагають зупинки навчального процесу і не придатні для вимірювання навантаження у реальному часі	Призначені для валідації об'єктивних сенсорних моделей; можуть бути використані після навчальних сесій
На основі продуктивності, що визначають рівень когнітивного навантаження із якості виконання завдань, точності і швидкості відповідей	Використовуються мультимодальні моделі машинного навчання (ML) для забезпечення високої точності класифікації	Виконується штучне вторинне завдання, що змінює умови навчання; ML-моделі функціонують як «чорний ящик» без можливості пояснення	Вимірювання навантаження в умовах подвійних завдань з перевірки правил та навичок водіння

Вимірювання діаметра зіниці (пупілометрія) є чутливим маркером когнітивного навантаження, доступним для вимірювання через стандартну веб-камеру з інфрачервоною підсвіткою [123].

Функціональна ближня інфрачервона спектроскопія дає можливість відстежити зміни рівня кисню у крові і є перспективним методом оцінювання когнітивного навантаження внаслідок мобільності й стійкості вимірювань при рухах людини [124].

Поведінкові методи не потребують спеціального обладнання і використовують дані щодо збільшення пауз перед відповіддю користувача, зміни траєкторії руху курсора, частоти використання підказок та розподілу часу між елементами завдання [125]. Айтрекінг (відстеження напрямку погляду людини та тривалості фіксацій) дозволяє оцінити розподіл уваги людини між елементами інтерфейсу, виділяючи ознаки перевантаження при звуженні зони сприйняття [126].

Суб'єктивні методи використовуються як ретроспективні для калібрування об'єктивних методів. Використовуються шкали NASA-TLX [127] та Пааса [128]. Перша є багатовимірною і, зокрема, враховує розумове і фізичне зусилля, а також часові обмеження і продуктивність. Друга використовує 9-бальну одновимірну оцінку.

Методи на основі продуктивності визначають навантаження основного завдання через вимірювання точності виконання вторинного завдання. При збільшенні когнітивного навантаження точність вторинної задачі зменшується [129]. В даному випадку зазвичай використовується мультимодальний підхід на основі машинного навчання, що підвищує точність класифікації рівнів когнітивного навантаження [130].

Персоніфікація виконується за результатами вимірювання когнітивного навантаження і включає статичну персоніфікацію, динамічне коригування складності навчального завдання, мультимодальну персоніфікацію, адаптивне мікронавчання, персоніфікацію з використанням носимих пристроїв. Характеристики цих підходів наведено у табл. 1.6.

Таблиця 1.6 – Підходи до персоніфікації за результатами вимірювання когнітивного навантаження

Підхід	Опис	Джерело даних про КН	Механізм адаптації	Обмеження
Статична персоніфікація	Одноразове визначення когнітивного профілю до сесії	Тести та опитувальники до навчання	Налаштування інтерфейсу НПС весь курс навчання	Не враховує динамічні зміни когнітивного навантаження упродовж сесії навчання
Динамічне коригування складності	Регулювання складності наступного завдання згідно поточного рівня когнітивного навантаження	Поведінкові сигнали	Зниження/підвищення складності завдання при перевантаженні / недовантаженні	Залежність від точності поведінкових сигналів
Мультимодальна персоніфікація	Враховує рівень знань та когнітивне навантаження при їх вивченні; використовується нейромережна архітектура	Поведінкові та фізіологічні сигнали	Адаптивне зважування, окремі ваги для новачків та окремі для досвідчених користувачів	Висока обчислювальна складність; потрібно калібрувати ваги для кожної групи
Адаптивне мікронавчання	Розбиття матеріалу на мікромодулі із динамічним вибором наступного модуля	Поточний рівень когнітивного навантаження і рівень засвоєння	При перевантаженні вибираються спрощені фрагменти	Потребує попереднього розбиття навчального матеріалу на мікромодулі
Персоніфікація з використанням носимих пристроїв	Безперервний моніторинг біометричної інформації через носимий сенсор	Фізіологічні сигнали в реальному часі	Класифікація когнітивного навантаження за дискретними рівнями складності для адаптації рівня складності	Залежність від якості пристроїв та готовності користувачів використовувати носимі пристрої

Статична персоніфікація передбачає визначення когнітивного профілю учня (ємності робочої пам'яті, типового стилю навчання, рівня тривожності) та використання цього профілю для налаштування інтерфейсу й вибору матеріалу до початку сесії навчання [131]. Підхід є інтерпретованим, однак ігнорує зміни когнітивного навантаження під час сесії навчання.

Підхід динамічного коригування складності навчання регулює складність наступного завдання згідно до поточного рівня когнітивного навантаження. При збільшенні навантаження складність завдання знижується, а при зменшенні – підвищується [132]. Проте підхід явно не враховує профіль користувача. Для реалізації підходу потрібен програмно-апаратний модуль, що відстежує стан користувача.

Підхід мультимодальної персоніфікації узагальнено враховує стан знань та когнітивне навантаження при вивченні цих знань з використанням адаптивного зважування пріоритетів. Для новачків більш високий пріоритет встановлюється для традиційного знання-орієнтованого навчання, а для досвідчених користувачів збільшується вага персоніфікованого компонента [133]. Підхід визначає поточний стан, без аналізу історії навчання.

Адаптивне мікронавчання розбиває навчальний курс на мікромодулі тривалістю декілька хвилин. Наступний мікромодуль вибирається на основі поточного когнітивного стану користувача та прогресу у навчанні. У випадку перевантаження використовуються роз'яснювальні компоненти [134]. Проте даний підхід також не враховує історію навчання.

Персоніфікація з використанням носимих пристроїв виконується на основі безперервного моніторингу біометричної інформації (варіабельність ритму серця, шкірно-гальванічна реакція, температура тіла) за допомогою носимого сенсора. Стан когнітивного навантаження користувача розбивається на дискретні рівні і використовується для адаптації швидкості навчання [135].

Розглянуті підходи відрізняються тим, що враховують поточне навантаження користувача, але не враховують історію його навчання. Ця історія відображає як рівень його знань, так і динаміку набуття знань.

Когнітивне навантаження при навчанні водінню має іншу специфіку порівняно з академічним навчанням, що визначає вимоги до персоніфікації при навчанні у автошколі. Ця специфіка обумовлена як складністю предметної галузі, так і особливостями пізнавальної діяльності під час оволодіння знаннями й навичками щодо керування автомобілем. Когнітивне навантаження відображає подвійну природу навчального завдання, фактор безпеки й тривожності, роботу у реальному часі. Наприклад, порядок проїзду регульованого світлофором перехрестя із трамвайною колією має бути віднесена до завдань із підвищеним рівнем складності.

Характеристики когнітивного навантаження при навчанні водінню наведено у табл. 1.7.

Подвійний характер навчального завдання виникає із-за доповнення когнітивного навантаження психомоторним навантаженням. Курсант автошколи паралельно обробляє сигнали у дорожній ситуації, правила дорожнього руху, показники приладів, а також виконує дії з керування автомобілем [129]. Дослідження щодо завдань з подвійним навантаженням підтверджують суттєве підвищення загального когнітивного навантаження та зниження ефективності аналізу дорожніх ситуацій новачком [129]. У досвідчених водіїв операції керування автомобілем виконуються в автоматичному режимі [117], що звільняє робочу пам'ять для аналізу дорожньої ситуації. У новачків ці операції виконуються контрольовано й самі по собі підвищують когнітивне навантаження, зменшуючи ресурс робочої пам'яті для засвоєння теоретичних знань [117]. Тому надання нового навчального матеріалу для учня має враховувати його поточне когнітивне навантаження, а не лише складність навчального матеріалу.

При академічному навчанні виникнення й виправлення помилки є типовим навчальним кроком. При водінні автомобіля помилка може мати негативні наслідки для водія, його пасажирів, а також інших учасників дорожнього руху. Критичність безпеки при водінні породжує тривожність при навчанні і підвищує когнітивне навантаження незалежно від складності завдання [136]. При іспиті у

МРЕВ або самостійних поїздок когнітивне навантаження курсанта автошколи зумовлене не лише складністю маршруту й дорожньої обстановки, а й станом тривоги, що визиває додаткове стороннє навантаження [136].

Таблиця 1.7 – Характеристики когнітивного навантаження при навчанні водінню

№	Причини підвищеного когнітивного навантаження	Особливість когнітивного навантаження	Відмінність від академічного навчання
1	Подвійний характер навчального завдання	Спільне когнітивне та психомоторне навантаження: одночасна обробка сигналів у дорожній ситуації та скоординовані моторні дії	При академічному навчанні психомоторна складова відсутня й завдання переважно когнітивні
2	Критичність безпеки	Критичність безпеки визиває тривожність при навчанні і підвищує когнітивне навантаження незалежно від складності завдання	В академічному навчанні помилка є допустимою й не приводить до незворотних наслідків
3	Трикомпонентна структура завдання	На когнітивне навантаження впливає кількість активних елементів у поточній дорожній ситуації, ступінь опанування теоретичними знаннями; рівень автоматизації базових моторних навичок майбутнього водія	В академічному навчанні когнітивне навантаження визначається лише складністю теми, а моторна та ситуативна складові відсутні
4	Відсутність довільної паузи у навчанні	Навчальну ситуацію не можна зупинити при перевантаженні як у реальному водінні так і при неперервному сценарії навчання	В академічному навчанні учень може зупинити, повернутись до навчального матеріалу у будь-який момент

Закінчення таблиці 1.7

№	Причини підвищеного когнітивного навантаження	Особливість когнітивного навантаження	Відмінність від академічного навчання
5	Асиметрія когнітивного навантаження курсанта автошколи та досвідченого водія	У новачка операції керування автомобілем виконуються контрольовано і генерують значне когнітивне навантаження, а у досвідченого водія ці операції автоматизовані	При традиційному навчанні відмінність між новачком та експертом проявляється у глибині знань, без фізіологічного навантаження

Персоніфікація має враховувати підвищене когнітивне навантаження при виборі складності навчальних сценаріїв.

У академічному навчанні когнітивне навантаження визначається складністю теми навчального завдання. При навчанні водінню навантаження залежить від трьох компонентів: складності дорожньої ситуації, ступеню опанування теоретичними знанням, рівнем автоматизації базових моторних навичок майбутнього водія. Складність ситуації залежить від кількості активних дорожніх елементів, включаючи поточний маневр, інтенсивність трафіку, кількість дорожніх знаків. Майбутній водій має використати правила пріоритету руху з одночасної координацією дій з управлінням авто. Наприклад, при виконанні лівого повороту у складній дорожній ситуації водій обробляє більше 8 елементів, що свідчить про високе когнітивне навантаження [116, 129]. Навчання з допустимим навантаженням призначене для формування знань щодо безпечного руху а також автоматизації навичок керування [117]. Персоніфікована система має враховувати трикомпонентну структуру завдання при теоретичному навчанні, знижуючи когнітивне навантаження в залежності від стану користувача шляхом підбору спрощеного навчального сценарію а також адаптацією інтерфейсу та збільшенням кількості повторів для автоматизації навичок.

При традиційному академічному навчанні учень має можливість зробити довільну паузу, призупинити перегляд лекції та повернутися до розгляду

складного фрагменту. У навчанні водінню курсант автошколи не має можливості призупинити вивчення навчального сценарію при настанні перевантаження [137]. Тому персоніфікована начальна система має бути проактивною, виявляти підвищений рівень когнітивного навантаження на основі поточного стану користувача та адаптувати складність начальних модулів до того моменту, коли настане когнітивне перевантаження [138]. Реактивна персоніфікація полягає у прийнятті рішення про корекцію навчального сценарію після перевантаження, тобто при реактивній персоніфікації рішення приймається із запізненням.

Асиметрія когнітивного навантаження курсанта автошколи та досвідченого водія виникає внаслідок різного рівня автоматизації моторних навичок. Дослідження [139] показує, що досвідчений водій використовує менше половини ресурсу робочої пам'яті на керування автомобілем. Інша половина пам'яті призначена для аналізу дорожньої ситуації. Новачок у такій же дорожній ситуації використовує майже всю робочу пам'ять. Тому новачок має суттєво вище когнітивне навантаження при виникненні непередбачуваних подій на дорозі. Така різниця підкреслює важливість оцінки поточного стану користувача і персоніфікації навантаження шляхом підбору навчальних сценаріїв.

Порівняння розглянутих підходів до персоніфікації дозволяє зробити висновок, що ці підходи не використовують єдине представлення для визначення рівня знань та стану користувача.

1.4 Постановка задачі дослідження

Аналіз методів підготовки водіїв з використанням навчальних програмних систем свідчить про обмеження традиційних підходів до персоніфікації та адаптації навчальних сценаріїв. Існуючі адаптивні системи навчання в автошколі використовують інформацію про рівень підготовки користувача для підбору навчальних модулів. Існуючі підходи до персоніфікації навчання враховують поточний стан когнітивного навантаження користувача навчальної програмної

системи проте не описують особливості навчання водінню, а також історію навчання і досягнутий при навчанні персональний рівень підготовки. Це приводить неврахування вимог персоніфікованої адаптації з використанням формалізованого опису профілю знань та поточного стану користувача при розробці архітектури програмної системи та методів персоніфікації й верифікації адаптивного процесу навчання.

Актуальність проблеми дисертаційного дослідження обумовлена необхідністю усунути невідповідність між практичною потребою персоніфікованої адаптації процесу підготовки водіїв з комплексним урахуванням як їх рівня знань так і поточного стану когнітивного навантаження та обмеженими можливостям існуючих підходів до адаптивного або персоніфікованого навчання.

Метою дисертаційного дослідження є розробка моделі та методів персоніфікованої адаптації процесу підготовки водіїв на базі гібридної архітектури програмної системи з використанням формалізованого представлення профілю знань та поточного стану користувача для підвищення ефективності навчання в програмних системах автошкіл.

Сформульовано наступні завдання дослідження відповідно до поставленої мети:

1. Аналіз стану проблеми адаптації та персоніфікації процесу навчання водіїв у автошколі.
2. Розробка моделі гібридної програмної архітектури з периферійними обчисленнями для персоніфікованої адаптації процесу підготовки в автошколі.
3. Розробка методів побудови вектору профілю користувача та вектору поточного стану користувача навчальної програмної системи.
4. Розробка методу персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача.
5. Розробка методу верифікації процесу вибору навчальних модулів з використанням технології інтелектуального аналізу процесів.

6. Реалізація гібридної програмної архітектури та експериментальна перевірка розроблених методів персоніфікованої адаптації процесу навчання.

Висновки до розділу 1

1. Аналіз адаптивного процесу навчання водіїв в автошколах показав, що такі процеси направлені на поступове набуття індивідуального досвіду в контрольованих умовах у відповідності до поточного рівня знань та навичок, причому ефективність підготовки водіїв значною мірою залежить від якісної інтеграції отриманих теоретичних знань і практичних навичок на основі контролю когнітивного навантаження. Останнє виникає внаслідок зовнішніх впливів а також при невідповідності навчального матеріалу когнітивним можливостям та рівню знань користувача.

2. Аналіз архітектур навчальних програмних систем засвідчив, що існуючі архітектурні рішення для навчальних програмних систем не враховують у архітектурі спільно рівень знань користувача та його поточний стан як два різних виміри процесу навчання, що взаємно доповнюють інформацію про користувача. Для підвищення ефективності навчання з урахуванням профілю користувача та його поточного когнітивного стану потрібно розробити гібридну програмну архітектуру, що поєднає можливості масштабованості мікросервісної архітектури для удосконалення процесу навчання та швидкої персоніфікованої адаптації на базі монолітної архітектури.

3. Дослідження підходів до персоніфікації навчання показує, що такі підходи визначають складність наступного навчального модуля або статично, до початку поточної сесії навчання, або динамічно, з урахуванням поточного стану користувача але без врахування обмежень щодо його рівня підготовки. Така персоніфікація може бути виконана з розбиттям сценарію навчання на мікромодулі, адаптивним встановленням пріоритетів за рівнем навантаження користувача, або на основі біометричної інформації від носимих пристроїв.

2 МОДЕЛІ ПІДТРИМКИ ПЕРСОНІФІКОВАНОГО АДАПТИВНОГО НАВЧАННЯ ВОДІЇВ У АВТОШКОЛАХ

2.1 Дескриптивне моделювання дорожньої ситуації для підтримки персоніфікованої адаптації процесу підготовки водіїв

Сучасні програмні системи, що використовуються для навчання водінню, використовують багаторівневий опис дорожніх сценаріїв [139, 140]. Такі системи представляють кожну дорожню ситуацію у вигляді фіксованої інтегральної сцени, тобто у вигляді єдиного об'єкта без відображення складності сцени як окремого параметру, наприклад у вигляді сцени «дорожнє перехрестя». Це означає, що у випадку, якщо потрібна менш або більш складна версія цього сценарію, розробники системи повинні створити новий об'єкт замість простої адаптації параметрів існуючого сценарію.

Також, існуючі системи не використовують біометричний зворотний зв'язок для персоніфікованої адаптації навчальних сценаріїв у режимі м'якого реального часу. Комерційні системи для навчання водінню коригують процес підготовки шляхом вибору із набору фіксованих сценаріїв і не застосовують адаптацію параметрів сцени під час навчання [141]. Складність апріорно задається для кожної сцени й тому поточне обчислення складності не виконується. Внаслідок цього існуючі системи не використовують механізм адаптації складності, що враховує як рівень підготовки, так і рівень когнітивного навантаження майбутнього водія.

Розроблений у даному підрозділі підхід до удосконалення існуючих моделей з використанням параметрів оцінки складності сценаріїв дозволяє [1]:

- представити сценарій навчання курсантів автошколи через комбінацію параметрів дорожньої ситуації, а не через фіксовану схему, як в існуючих системах;

- отримати числову оцінку складності сценарію навчання;

– адаптувати сценарій у режимі м'якого реального часу на основі порівняння складності дорожньої сцени дорожньої сцени та варіабельності серцевого ритму курсанта автошколи.

Запропонована адаптивна модель дорожньої ситуації включає такі складові:

- елементи дорожньої ситуації Sd , що включають дорожні об'єкти, зокрема дорожні знаки, пішоходів, транспорт, світлофори;
- правила дорожнього руху Pr ;
- дії Act у дорожній ситуації;
- набір оцінок складності L з використанням відповідних параметрів складності дорожньої ситуації;
- множину Gs навчальних сценаріїв, які мають враховувати складність навчання на основі даних від біометричного зворотного зв'язку.

Формально адаптивна модель дорожньої ситуації Ms має вигляд:

$$Ms = \langle Sd, Pr, Act, L, Gs \rangle. \quad (2.1)$$

Зв'язки між рівнями Sd елементів дорожньої ситуації задаються множиною відношень R , що визначають структурні та регуляторні ролі. Підмножина структурних ролей R_s визначає постійні зв'язки між елементами дорожньої ситуації, наприклад наявність дорожніх знаків, транспортних засобів, пішоходів в дорожній ситуації. Підмножина регуляторних ролей R_r задає відношення між правилами, елементами ситуацій та діями, тобто визначають умови застосування правила для конкретного елемента в дорожній ситуації і наступні дії після застосування правила.

Для опису елементів моделі Ms та відношень R використовується дискрипційна логіка, яка задає концепти (класи), що відображають множини об'єктів певного типу у предметній області, ролі, що визначають бінарні відношення між цими об'єктами, та екземпляри, що представляють конкретні об'єкти.

Дорожня ситуація характеризується таким базовим набором концептів, що відображають елементи *Sd*, правила *Pr*, дії *Act* та визначаються в рамках універсального концепту • :

$$(\text{RoadSituation}, \text{TrafficSign}, \text{Vehicle}, \text{Pedestrian}, \text{Rule}, \text{Action}) \Phi \bullet, \quad (2.2)$$

де *RoadSituation* – концепт дорожньої ситуації;

TrafficSign – концепт дорожнього знаку;

Vehicle – концепт транспортного засобу;

Rule – концепт правила дорожнього руху;

Pedestrian – концепт пішохода;

Action – дія водія транспортного засобу.

Аналогічно, для концепту транспортних засобів та концепту дорожніх знаків задаються підкласи легковий автомобіль (*Car*), вантажний автомобіль (*Truck*), мотоцикл (*Motorcycle*), автобус (*Bus*):

$$(\text{Car}, \text{Truck}, \text{Motorcycle}, \text{Bus}) \Phi \text{Vehicle}. \quad (2.3)$$

Для класу дорожніх знаків виділяються підкласи попереджувальних знаків (*WarningSign*), знаків пріоритету (*PrioritySign*), заборонних знаків (*ProhibitionSign*), наказових знаків (*MandatorySign*), спеціальних (*SpecialOrderSign*), інформаційно-вказівних знаків (*InformationSign*), сервісу (*ServiceSign*):

$$\left(\begin{array}{l} \text{WarningSign}, \text{PrioritySign}, \text{ProhibitionSign}, \\ \text{MandatorySign}, \text{SpecialOrderSign}, \\ \text{InformationSign}, \text{ServiceSign} \end{array} \right) \Phi \text{Vehicle}. \quad (2.4)$$

Структурні ролі, такі як «містить елемент» (*hasElement*), «містить знак» (*hasSign*), «містить транспортний засіб» (*hasVehicle*), «включає пішохода» (*hasPedestrian*) визначають структуру дорожньої ситуації, наприклад:

$$hasVehicle : Situation \times Vehicle. \quad (2.5)$$

Регуляторні ролі встановлюють відношення між правилами та ситуаціями, зокрема застосування правила до ситуації (*appliesTo*), обмеження для транспортного засобу (*restricts*), вимога дії (*requires*), порушення правил (*violates*).

Наприклад, у випадку дорожнього знаку «Дай дорогу» водій має зробити зупинку перед перехрестям згідно правила 7.8 ПДР, яке й вимагає від водія дій «Зупинка перед перехрестям». Така ситуація у моделі задається за допомогою наступних концептів:

$$Concepts = \left\{ \begin{array}{l} TrafficSign, Rule, Action, \\ GiveWaySign \text{ ф } TrafficSign, \\ StopAction \text{ ф } Action. \end{array} \right\}, \quad (2.6)$$

де *GiveWaySign* – знак «Дай дорогу»;

StopAction – дія зупинки транспортного засобу.

Тоді клас ситуацій «Дай дорогу (*GiveWaySituation*)» виникає у випадку, якщо клас дорожніх ситуацій (*RoadSituation*) має елемент типу знак «Дай дорогу (*GiveWaySign*)», що відображається за допомогою структурної ролі *hasSign*. Ця роль пов'язує дорожню ситуацію та дорожній знак і визначає множину всіх ситуацій (*RoadSituation*), в яких є знак «Дай дорогу (*GiveWaySign*)»:

$$\begin{aligned} GiveWaySituation &\equiv \\ &\equiv RoadSituation \text{ у } (\exists hasSign.GiveWaySign). \end{aligned} \quad (2.7)$$

Загальна послідовність активації у дорожній ситуації, згідно якої знак запускає правило, а правило запускає дію, має такий вигляд:

$$\begin{aligned} \text{Activation} &\equiv \\ &\equiv \text{TrafficSign } y \left(\exists \text{appliesTo.} \left(\text{Rule } y \left(\exists \text{requires.Action} \right) \right) \right). \end{aligned} \quad (2.8)$$

Роль *appliesTo* визначає потребу застосувати правило поточної ситуації, тобто виділяє підмножину дорожніх ситуацій, в яких правило *Rule* застосовується. Перетини із *TrafficSign* обмежує підмножину ситуацій лише заданим типом знаків, до яких правило має бути застосовано.

Послідовність активації правила та дії у даній ситуації враховує наявність знака «Стоп (*StopAction*)»:

$$\begin{aligned} \text{GiveWayActivation} &\equiv \\ &\text{GiveWaySign } y \exists \text{appliesTo.} \left(\text{Rule } y \exists \text{requires.StopAction} \right). \end{aligned} \quad (2.9)$$

Наприклад, для дорожніх знаків такий зв'язок може мати вигляд: Дорожній знак – has_rule – «Уступи дорогу» – requires_action – «Зупинка перед перехрестям», що забезпечує умови для автоматизації виявлення суперечностей між правилами та перевірки коректності сценаріїв.

Оцінка сценарію вивчення дорожньої ситуації орієнтована на підтримку заданого рівня когнітивного навантаження курсанта автошколи і тому ґрунтується на теорії когнітивного навантаження [56], яка класифікує види когнітивного навантаження. Вибір атрибутів для оцінки складності дорожніх ситуацій базується на методології оцінки робочого навантаження NASA-TLX, яка визначає шість вимірів когнітивного та фізичного попиту [137], а також на проведених дослідженнях щодо навчання водінню, в яких розглядався зв'язок між когнітивним навантаженням та параметрами ритму серця [142, 143].

На основі аналізу вказаних досліджень запропоновано такі параметри для оцінки дорожньої ситуації, які мають визначальний вплив на когнітивне навантаження курсанта:

- кількість об'єктів у складі дорожньої ситуації As_1 ;
- час на прийняття рішення щодо дії у дорожній ситуації As_2 ;
- кількість варіантів можливих дій, які курсант може виконати у поточній дорожній ситуації As_3 .

Перший параметр дорожньої ситуації вибрано відповідно до теорії обмеженої ємності уваги та концепції «вузького місця уваги» [56], згідно яких людський мозок ефективно може паралельно обробляти обмежену кількість об'єктів з достатньо високим рівнем деталізації. Збільшення кількості об'єктів вище вказаної межі приводить до експоненціального зниження якості аналізу кожного окремого об'єкта. Дослідження залежностей між станом та рівнем навантаження водія виявили кореляційний зв'язок між порушеннями варіабельності серцевого ритму (HRV) та кількістю транспортних засобів, що бачить водій. Водіння на шосе з мінімальною кількістю об'єктів у полі зору характеризується базовим рівнем HRV, а для міського трафіку з високою кількістю об'єктів характерним є зниження HRV [143].

Для даного параметра розроблено 5 градацій значень, щоб відобразити повний діапазон реальних дорожніх ситуацій, що відповідають простій навчальній ситуації, незавантаженій міській обстановці, типовому міському трафіку, інтенсивному міському руху та трафіку з максимальною кількістю об'єктів. Перша градація дає можливість курсанту автошколи зосередитися лише на основних навичках керування. Така ситуація забезпечує навчання водінню без суттєвого перенавантаження когнітивної системи майбутнього водія, оскільки згідно дослідженню [144] водії – новачки вже при 1-2 автомобілях на дорозі виявляють ознаки стресу.

Друга градація відповідає міській обстановці на невеличких вуличках, коли майбутній водій обробляє інформацію про множину об'єктів в комфортних часових рамках.

Третя градація відображає типовий міський рух середньої інтенсивності, який потребує від курсанта навичок вибору та пріоритизації об'єктів на дорозі за рівнем небезпеки.

Четверта градація моделює міський рух на інтенсивно завантажених вулицях, який вимагає від курсанта швидкої обробки інформації для постійного переключення уваги в межах видимої множини автомашин.

Остання градація кількості автомашин моделює критичну ситуацію на дорозі, яка може перевантажити можливості когнітивної системи людини обробляти просторову інформацію.

Вибір параметру інтервалу часу на прийняття рішення в дорожній ситуації базується на розробленій NASA методиці NASA-TLX суб'єктивної оцінки когнітивного навантаження. Дана методика дає можливість визначити, наскільки складним сприймається виконання певного завдання людиною [127]. Методика використовує 6 показників, ключовим з яких є ментальна напруженість (Mental Demand). Інтервал часу на прийняття рішення в дорожній ситуації відображає показник ментальної напруженості, оскільки час, потрібний для прийняття рішення, зростає логарифмічно в залежності від кількості варіантів рішення [145]. Дослідження зв'язку варіабельності серцевого ритму та ментальної напруженості в адаптивних системах навчання показали високе значення кореляції, яке є найвищим серед всіх шести компонентів навантаження за методикою NASA-TLX [127].

Для даного параметру вибрано 4 рівня градації, що охоплюють весь діапазон реакцій водія: починаючи з екстреної реакції й до виконання планового маневру, коли водій має достатньо часу для аналізу дорожньої ситуації. Градація 1 представляє критичну ситуацію, яка потребує швидкої, на рівні автоматизму, реакції на основі вже сформованих алгоритмів дій. Градація 2 даного параметру відображає потребу у швидкому прийнятті рішення у динамічній дорожній ситуації, коли курсант автошколи встигає виконати базовий аналіз обстановки, але має недостатньо часу для детального розгляду всіх альтернатив, що потребує навичок вибору пріоритетів і оцінки ризиків. Градація 3 відображає типове

маневрування у дорожній ситуації, коли курсант має достатньо часу для аналізу поточної обстановки, подальшої оцінки можливих альтернативних дій, а також вибору раціонального рішення. Градація 4 забезпечує достатній комфортний інтервал часу для водіїв-новачків, який дає можливість повністю задіяти робочу пам'ять людини (тривалість в межах 20-30 секунд [146]) та повністю обміркувати майбутній маневр. Дана градація призначена для початкових етапів навчання водінню, коли майбутній водій ще не досяг рівня автоматизму для навичок прийняття рішень.

Параметр кількості альтернативних дій, які водій має розглядати в якості потенційних відповідей у поточній ситуації, безпосередньо відображає складність процесу прийняття рішення. Даний параметр згідно теорії когнітивного навантаження репрезентує когнітивне навантаження, яке пов'язано із побудовою схематичних знань [56]. Обґрунтування цього вибору ґрунтується на емпіричних дослідженнях, які демонструють, що кожна додаткова альтернатива дії збільшує Mental Demand на 8-12% та посилює порушення варіабельності серцевого ритму [127]. На відміну від щільності об'єктів, яка характеризує зовнішню складність ситуації, варіанти дій репрезентують внутрішню складність прийняття рішення та вимагають від водія розвитку навичок аналізу множинних сценаріїв та моделювання наслідків кожного потенційного маневру [139].

Оцінка рівня складності дорожньої ситуації виконується на основі згортки значень as_{ij} цих параметрів:

$$L = \sum_{i,j} w_i as_{ij}, (\forall i = \overline{1,3}) as_{ij} \in As_i, \quad (2.10)$$

де ваги w_i кожної групи параметрів визначаються на основі опитування курсантів автошколи.

Таким чином, розроблена модель дає можливість адаптивно описати дорожні ситуації, що використовуються при навчанні в автошколі, а також виконати оцінку рівня складності цих ситуацій.

2.2 Моделювання процесу обробки біометричних даних у програмних системах навчання водіїв в автошколах

Ефективність підготовки водіїв з використанням навчальних програмних систем суттєвою мірою залежить від можливості адаптувати складність сценаріїв підготовки до індивідуальних когнітивних можливостей та стану курсанта автошколи. Існуючі системи використовують типові навчальні програми, які не враховують поточний стан користувачів, що може привести до когнітивного перевантаження користувачів, які мають обмежений досвід, а також до недовантаження досвідчених курсантів. Такі обмеження приводять до суттєвого зниження ефективності навчального процесу.

Таким чином, підвищення ефективності підготовки водіїв потребує вирішення задачі персоніфікації процесу навчання з урахуванням когнітивного навантаження курсантів. Для реалізації цієї задачі необхідно обґрунтувати та розробити модель процесу підтримки зворотного зв'язку від курсанта в навчальній програмній системі на основі обробки біометричних даних з тим, щоб забезпечити динамічну персоніфікацію навчальних сценаріїв для таких користувачів. Оцінка поточного стану курсанта автошколи потребує постійного вимірювання фізіологічних реакцій на навчальній сесії. Точність цього вимірювання впливає на оцінку стану користувача при навчанні і на підбір навчальних модулів. Стан користувача зазвичай вимірюється з урахуванням варіабельності та частоти серцевих скорочень, а також рівня його когнітивного навантаження.

Існуючі підходи до біометричної оцінки поточного стану користувача базуються на різних фізичних схемах вимірювання (накладні електроди, нагрудні електрокардіографи, фотоплетизмографія (photoplethysmography,

PPG)), що відрізняються за точністю та можливістю використання в умовах автошкіл.

Накладні електроди, що закріплені на тілі, використовуються в лабораторному медичному обладнанні, наприклад, полісомнографах та холтерівських моніторах (Holter monitors). Полісомнограф одночасно може фіксувати до двадцяти фізіологічних сигналів, наприклад електрокардіограму, електроенцефалограму, а також міограму та реєстрацію дихання. Даний прилад забезпечує високу точність. Холтерівський монітор є портативним прибором з накладними електродами. Накладні електроди розміщують у на тілі людини у кількох точках для запису ЕКГ. Пристрій носить упродовж заданого часу для запису історії вимірювань. Даний підхід має обмеження для використання при навчанні в автошколі, оскільки електроди розміщує кваліфікований медичний персонал, який попередньо готує пацієнта.

Нагрудні мобільні електрокардіографи є компактною реалізацією підходу до вимірювання з накладними електродами. Вони використовують тканинний пояс із вбудованими текстильними електродами. Цей пояс охоплює грудну клітку і реєструє електричний потенціал серцевого м'яза. Даний прилад забезпечує високу точність вимірювань, близьку до попереднього підходу та без потреби у медичному персоналі. Проте підхід має фізичні обмеження. Нагрудний пояс стискає грудну клітку при носінні, містить кабелі і тому несумісний із ефективним навчанням.

Споживчі смарт-годинники не використовують електроди, а використовують оптичний датчик на внутрішній поверхні годинника (метод фотоплетизмографії, ФПГ). Оптичний датчик випромінює світло у зелено-червоному діапазоні. Вбудований фотодетектор вимірює інтенсивність відбитого від шкіри сигналу. Об'єм крові в капілярах коливається синхронно зі скороченнями серця, то світло на вході фотодетектора пульсує з такою ж частотою. Ця пульсація світлового сигналу містить інформацію про серцевий ритм. Також носіння годинника не накладає обмежень на рухи людини при навчанні. Водночас оптичний підхід до вимірювань є більш чутливим до рухів

людини, ніж електричний. Також споживчі смарт-годинники не мають стандартизованої і моделі збору даних.

Формування множини вимог до пристрою біометричної оцінки поточного стану користувача базується на аналізі контексту як набору умов, у яких цей пристрій має функціонувати в програмній навчальній системі. Кожна вимога має відображати обмеження або потребу у застосуванні пристрою. Такий підхід реалізує методологію контекстно-залежного проєктування вимог.

Операційний контекст визначають чотири чинники.

По-перше, біометричний сигнал повинен містити всю необхідну інформацію для розрахунку вектору поточного стану користувача програмної системи, в тому числі інформацію щодо варіабельності серцевого ритму. Обчислення останнього визначає обмеження щодо частоти дискретизації.

По-друге, пристрій біометричних вимірювань функціонує в умовах навчання, коли користувач може виконувати рухи, що обмежує форм-фактор пристрою і не допускає його закріплення на тулубі людини.

По-третє, передача біометричних даних реалізує критичний шлях обробки тому має бути забезпечена мінімальна затримка передачі даних від пристрою до вузла обчислень, яка забезпечила б роботу в режимі м'якого реального часу.

В-четвертих, програмна система має розгортатись в автошколах, що обмежує вартість пристрою.

Сукупність контекстних чинників визначає наведені у табл. 2.1 вимоги до пристрою отримання біометричних даних. Більшість розглянутих пристроїв не задовольняють вимогам автономності, можливості носіння та обмеження по ціні. Серед проаналізованих пристроїв сукупності вимог задовольняють лише смарт-годинники.

Модель процесу обробки біометричних даних M_{bio} складається з 5 фаз:

$$M_{bio} = \langle \Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5 \rangle, \quad (2.11)$$

де Φ_1 – фаза отримання біометричних даних;

Φ_2 – фаза попередньої обробки даних;

Φ_3 – фаза відбору первинних ознак;

Φ_4 – фаза формування формального опису поточного стану користувача;

Φ_5 – фаза передачу опису поточного стану користувача до модулю персоніфікованої адаптації процесу навчання.

Таблиця 2.1 – Вимоги до пристрою вимірювання біометричних даних

Вимога	Контекстний чинник
1. Пристрій для носіння	Пристрій не обмежує рухи користувача програмної системи навчання
2. Бездротова передача даних у режимі м'якого реального часу	Передача біометричних даних реалізує критичний шлях обробки тому має бути забезпечена мінімальна затримка передачі даних від пристрою до вузла обчислень
3. Автономність – декілька годин моніторингу (робочий день)	Пристрій біометричних вимірювань функціонує під час навчання, що складає декілька годин
4. Невисока ціна пристрою на рівні масової продукції	Вартість пристрою є критичним фактором для прийняття рішення щодо використання пристроїв для навчання
5. Наявність високочастотного ФПГ(<i>PPG</i>) - сенсора	Високочастотна обробка (наприклад, 60 циклів на хвилину) забезпечує роботу в режимі м'якого реального часу
6. Наявність програмного інтерфейсу	Передача біометричних даних в центр обробки з використанням програмного інтерфейсу

Сукупність цих фаз відповідає послідовності перетворень даних від датчика до фінального структурованого опису поточного стану користувача навчальної програмної системи [8].

Загальну схему процесу обробки біометричних даних користувача представлено на рис. 2.1.

Фаза Φ_1 є комунікаційною. Передача даних здійснюється без їх обробки за протоколом Bluetooth Low Energy.

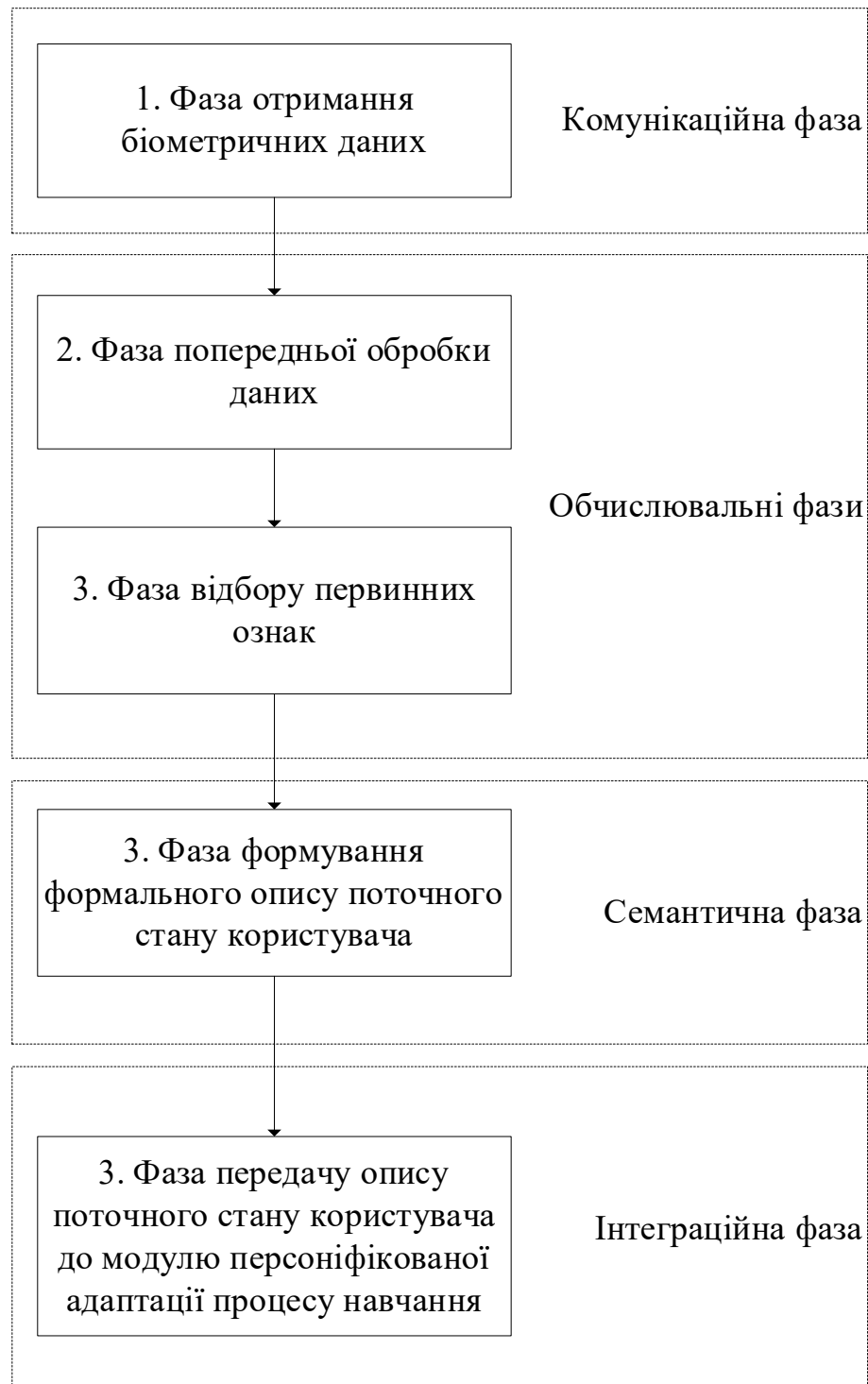


Рисунок 2.1 – Схема процесу обробки біометричних даних користувача

Фази Φ_2, Φ_3 є обчислювальними і призначені для підготовки даних до опису стану користувача. Фаза Φ_4 забезпечує перетворення вхідного сигналу на компоненти вектору, що визначають поточний стан користувача. Фаза Φ_5 забезпечує інтеграцію модулів програмної системи. Структурований опис стану користувача передається для персоніфікованої адаптації процесу підготовки

водіїв. Кожна з фаз має вхідний і вихідний інтерфейс, що дає можливість змінювати дії на будь-якій фазі незалежно від інших складових процесу обробки біометричних даних.

Необхідність фаз Φ_2, Φ_3 є наслідком того, що необроблений сигнал смарт-годинника містить не лише корисну, а й шумову складові і не може бути безпосередньо використаний для обчислення властивостей серцевого ритму.

ФПГ датчика смарт-годинника не вимірює безпосередньо серцевий ритм. Він вимірює зміну інтенсивності відбитого світла, яка залежить від пульсації серця, положення датчику на руці та електричних перешкод. Серце пульсує і проштовхує кров у капіляри. Відображення цієї пульсації є корисним сигналом, який використовується для побудови опису стану користувача. Проте користувач, як правило, робить рухи руками при навчанні. Ці рухи приводять до зміщення датчика відносно руки і зміни інтенсивності відбитого світла незалежно від серцевого ритму. Також сам сигнал фотодатчика містить шумову складову. Тому сигнал $x(t)$ датчика смарт-годинника у довільний момент часу t має вигляд:

$$x(t) = x_{\text{cardio}}(t) + x_{\text{motion}}(t) + x_{\text{el}}(t), \quad (2.12)$$

де $x_{\text{cardio}}(t)$ – компонента сигналу, частота якої лежить у діапазоні значень ритму серця людини;

$x_{\text{motion}}(t)$ – шум внаслідок, що виникає при рухах зап'ястя у процесі навчання;

$x_{\text{el}}(t)$ – шумова складова сигналу фотодатчика.

Можливість розділення корисного та шумових сигналів у (2.12) оснований на тому, що складові сигналу датчика $x_{\text{cardio}}(t)$, $x_{\text{motion}}(t)$ та $x_{\text{el}}(t)$ розділені по частоті.

Складова $x_{\text{cardio}}(t)$ знаходиться у діапазоні частот ритму серця людини, складова $x_{\text{motion}}(t)$ є низькочастотною, а $x_{\text{el}}(t)$ здебільшого містить

високочастотний сигнал. Дана властивість дозволяє відфільтрувати шумові складові і виділити кардіосигнал.

Така фільтрація виконується на фазі Φ_2 процесу обробки біометричних даних. Послідовність фільтрації містить наступні кроки.

Крок 2.1. Фільтрація вхідного сигналу в діапазоні від 0,5 Гц до 4 Гц.

Фільтрація дає можливість виділити сигнал серцебиття, який, як правило, знаходиться у діапазоні від 0,8 до 2 Гц, що відповідає пульсу від 48 до 120 ударів на хвилину. Для фільтрації використовується смугово-пропускний фільтр четвертого порядку, який ослаблює сигнали за межами діапазону серцебиття на 24 Дб.

Крок 2.2. Детектування піків серцебиття.

Детектування піків R_j включає диференціювання вхідного сигналу для визначення зміни сигналу від датчика, зведення в квадрат для підсилення різниці між піками сигналу та шуму, а також знаходження піку сигналу, що перевищує адаптивний поріг. Використовується ковзне вікно. Адаптивний поріг визначається як середнє значення між піком сигналу і шумом.

Крок 2.3. Обчислення інтервалів NN_j між ударами серця:

$$NN_j = R_{j+1} - R_j, j = 1, 2, \dots, J - 1, \quad (2.13)$$

де R_j – піковий сигнал, що отриманий на попередньому кроці;

J – кількість піків.

Крок 2.4. Фільтрація викидів.

Виключаються значення інтервалів NN_j , для яких відносне відхилення від медіанного значення перевищує поріг ε_{NN} . Умова фільтрації має вигляд:

$$\frac{|NN_j - NN|}{NN} > \varepsilon_{NN}, \quad (2.14)$$

де NN – медіанне значення інтервалу між ударами серця.

Фаза Φ_3 містить кроки відбору первинних ознак.

Крок 3.1. Обчислення показника $RMSSD$ – середньоквадратичного відхилення послідовних різниць:

$$RMSSD_{NN} = \sqrt{\frac{1}{J-1} \sum_{j=1}^{J-1} (NN_{j+1} - NN_j)^2}. \quad (2.15)$$

Даний показник вимірює мікроколивання між ударами серця.

Крок 3.2. Розрахунок частоти серцевих скорочень.

Розрахунок виконується на інтервалі $T = 60000 \text{ мс}$:

$$HR = \frac{T}{\overline{NN}}, \quad (2.16)$$

де $\overline{NN}$ – середнє значення інтервалу між ударами серця в мікросекундах.

Крок 3.3. Обчислення нормалізованої варіабельності серцевих скорочень HRV_{norm} :

$$HRV_{norm} = \frac{RMSSD - RMSSD_{min}}{\Delta RMSSD}, \quad (2.17)$$

де $\Delta RMSSD = RMSSD_{max} - RMSSD_{min}$ – максимальний діапазон варіабельності;

$RMSSD_{max}, RMSSD_{min}$ – максимальне й мінімальне значення варіабельності серцевих скорочень.

Значення HRV є первинною ознакою поточного стану людини. Це значення знижується при перенавантаженні користувача.

Крок 3.4. Розрахунок нормалізованої частоти серцевих скорочень HR_{norm} :

$$HR_{norm} = \frac{HR - HR_{min}}{HR_{max} - HR_{min}}, \quad (2.18)$$

де HR_{min} , HR_{max} – мінімальні і максимальні значення серцевих скорочень, що отримані за формулою (2.16).

Комбінація частоти та варіабельності серцевих скорочень відображає стан користувача. Наприклад, поєднання підвищеної частоти серцевих скорочень і зниженої варіабельності є патерном стану перенавантаження користувача навчальної системи.

Розроблена модель процесу обробки біометричних даних визначає фази перетворень від ФПГ сигналу смарт-годинника до опису поточного стану користувача. Модель відповідає наведеним у табл. 2.1 вимогам до вимірювання таких даних у процесі навчання з урахуванням рухів користувача навчальної системи та потреб практичного використання щодо автономності й невисокої вартості.

2.3 Модель гібридної програмної архітектури з периферійними обчисленнями для персоніфікованої адаптації процесу підготовки в автошколі

Проведений у першому розділі аналіз архітектурних підходів до розробки програмних систем показав, що базові архітектури не забезпечують задоволення суперечливих нефункціональних вимог до адаптивної програмної системи навчання в автошколі. Така система має забезпечити низькі затримки при персоніфікованій адаптації за умови обмеженої вартості володіння та конфіденційності біометричних даних.

Монолітна багат шарова архітектура забезпечує зменшення затримок при обробці даних, оскільки всі компоненти системи працюють як один обчислювальний процес на комп'ютері користувача. Проте така архітектура потребує використання достатньо дорогого комп'ютерного обладнання на кожному навчальному місці, що обмежує масштабування.

Клієнт-серверна архітектура використовує централізовані обчислення, що зменшує витрати на клієнтські модулі, проте має суттєву затримку часу через використання глобальної мережі. Загальна затримка, як правило, помітно перевищує допустиму межу для систем реального часу.

Програмна архітектура з периферійними обчисленнями включає сервер разом користувачем, що суттєво знижує затримку в мережі. Однак такий монолітний периферійний сервер має обмеження масштабування при зростанні кількості одночасних користувачів.

Мікросервісна архітектура полегшує масштабування та розгортання, але на практиці передбачає розміщення значної частини логіки у хмарі, що ускладнює виконання вимоги до затримки при адаптації а також вимоги конфіденційності біометричних даних.

Даний підрозділ присвячений розробці гібридної програмної архітектури навчальної програмної системи, що використовує переваги розглянутих архітектур та містить три рівні: користувача; периферійний; мікросервісний. Розподіл модулів по рівням виконується з урахуванням нефункціональних вимог згідно підходу, описаного в специфікації Architecture Analysis & Design Language [147], що розроблена для авіоніки, а також автомобільних та автономних систем з критичними нефункціональними вимогами, зокрема щодо часу відклику системи, безпеки тощо. Такі вимоги мають бути вбудовані в модель як її властивості [6].

Розроблена архітектура є гібридною, оскільки об'єднує три структурно різні патерни взаємодії між модулями програмної системи [2]:

- синхронна обробку в межах одного обчислювального процесу на периферійному сервері;
- синхронна взаємодія між клієнтом і периферійним сервером в локальній мережі;
- асинхронна взаємодія на основі подій між периферійним і мікросервісним рівнями з використанням черги повідомлень.

Кожен із цих шаблонів призначений для задоволення відповідних нефункціональних вимог, а їх поєднання дає можливість виконати у сукупності вимоги до затримки, вартості та конфіденційності даних.

Розроблена гібридна архітектура містить користувацький, периферійний та мікросервісний рівні:

$$M_{Arch} = \langle U, D, S, R \rangle, \quad (2.19)$$

де U – набір модулів рівня користувача;

E – множина модулів периферійного рівня;

S – модулі мікросервісного рівня, що реалізують логіку навчання;

R – множина функціональних та нефункціональних вимог.

На користувацькому рівні реалізуються функції відображення дорожньої обстановки для користувача та збору біометричних даних. На периферійному рівні реалізовано алгоритми персоніфікованої адаптації. На мікросервісному рівні, в хмарі, реалізується верифікація та аналітика, оскільки при вирішенні цих задач часові обмеження не є критичними.

Розподіл модулів навчальної програмної системи між користувацьким, периферійним та мікросервісним рівнями визначається критеріями затримки $K_{latency}$, витрат K_{cost} та ізоляції K_{isol} мікросервісного рівня від критичного шляху обробки біометричних даних користувача для персоніфікованої адаптації процесу навчання.

Затримка на критичному шляху визначається як сумарний час збору біометричних даних, роботи модулів периферійного рівня та мережових затримок у локальній мережі:

$$K_{latency} : (Lt_U + Lt_D + Lt_{U-D} + Lt_{D-U}) \leq Lt_{threshold}, \quad (2.20)$$

де Lt_U – затримка при зборі біометричних даних від користувача;

Lt_D – затримка на обробку даних на периферійному рівні;

$Lt_{U-D} + Lt_{D-U}$ – затримка на передачу даних в локальній мережі між користувацьким та периферійним рівнями.

Порогове значення $Lt_{threshold}$ має бути таким, щоб користувач не відчував затримки при роботі з навчальним модулем. Наприклад, порогове значення 0,25 секунди не викликає відчуття затримки при відображенні навчального модуля користувачеві.

Критерій витрат K_{cost} визначає вартість гібридної архітектури в залежності від кількості користувачів та рівня завантаження периферійного сервера:

$$K_{cost} : \left(nC_U + \left\lceil \frac{n}{h} \right\rceil C_D + C_S \right) \leq C_{threshold}, \quad (2.21)$$

де n – кількість курсантів в автошколі

h – кількість користувачів, що використовують сервер;

$\left\lceil \frac{n}{h} \right\rceil$ – функція стелі, наприклад $2 > \frac{n}{h} > 1 \Rightarrow \left\lceil \frac{n}{h} \right\rceil = 2$;

C_U – вартість комп'ютера для навчання користувача;

C_D – вартість периферійного сервера;

C_S – вартість хмарного сервісу.

Критерій K_{isol} дає можливість оцінити ізоляцію рівня мікросервісів від шляху обробки біометричних даних користувача, оскільки останній є критичним шляхом. Тобто в послідовності «збір біометричних даних–класифікація стану користувача – персоніфікована адаптація навчального модуля» не має бути синхронного виклику до мікросервісного рівня, тому що такий виклик приводить до затримок при персоніфікованій адаптації процесу навчання.

Такий підхід реалізується за допомогою асинхронної черги подій. Дана черга реалізується таки чином. Периферійний сервер публікує події. Мікросервіси підписуються на топіки з потрібними подіями, після чого

обробляють ці події у фоновому режимі. Тому затримки обробки на мікросервісному рівні не враховуються у (2.20).

Користувацький рівень містить модулі взаємодії з користувачем $U = \{u_1, u_2, u_3, u_4\}$: u_1 – модуль навчального інтерфейсу з користувачем; u_2 – модуль обробки біометричних даних користувача; u_3 – модуль обробки вхідних даних від користувача; u_4 – модуль взаємодії з периферійним сервером.

Модуль навчального інтерфейсу підтримує взаємодію із користувачем і забезпечує оперативну візуальну відповідь на дії користувача. Даний модуль отримує параметри персоніфікованої адаптації для вибору навчального модуля. Ці параметри задають тип і рівень складності навчального матеріалу так, щоб несуттєво перевищити поточний рівень знань користувача і забезпечити мотивацію до навчання без перенавантаження курсанта. Модуль відображає для користувача готовий екран навчання і меню взаємодії. Такий екран може містити:

- тестові запитання щодо Правил дорожнього руху;
- мультимедійні ілюстрації навчальних дорожніх ситуацій;
- інформацію про результати відповідей курсанта.

Модуль обробки біометричних даних користувача накопичує дані біометричного моніторингу для визначення поточного стану користувача навчальної програмної системи. Даний модуль виконує:

- постійне вимірювання варіабельності серцевого ритму з використанням зв'язку з носимим пристроєм (смарт-годинником);
- вимірювання з використанням ковзного вікна;
- формування вектору поточного стану користувача навчальної системи;

Отримані дані дають можливість оцінити наявність стресу у користувача, а також рівень його поточного когнітивного навантаження. Тому даний модуль повинен надавати вектор стан користувача з мінімальною затримкою.

Модуль обробки вхідних даних від користувача перетворює події у узгоджений формат, який у подальшому використовується модулем диспетчеризації. Модуль обробки вхідних враховує події, які виникли внаслідок

дій користувача (натискання клавіш на клавіатурі, рухи миші) та формує структурований опис цих подій, який аналізується на рівні периферійного сервера. Затримки в даному модулі є несуттєвими і не впливають на критичний шлях, пов'язаний із обробкою біометричних даних.

Модуль взаємодії з периферійним сервером виконує передачу даних на периферійний сервер, враховуючи нефункціональні вимоги щодо затримок та конфіденційності біометричних даних. Він передає вектор стану користувача на периферійний сервер і приймає від сервера параметри персоніфікованої адаптації з використання віддаленого виклику процедур, що дає можливість передати дані з невеликою затримкою.

Периферійний рівень реалізує персоніфіковану адаптацію процесу навчання на основі використання методів формування профілю користувача, методу персоніфікованої адаптації навчальних модулів та методу верифікації адаптивної програмної системи, які детально розглянуті в розділі 3. Даний рівень включає модулі $D = \{d_1, d_2, d_3, d_4, d_5\}$: d_1 – модуль формування профілю користувача; d_2 – модуль визначення поточного стану користувача на основі біометричних даних; d_3 – модуль диспетчеризації навчальних модулів; d_4 – модуль навчального контенту; d_5 – модуль моніторингу нефункціональних характеристик.

Модуль формування профілю користувача формує вектор, що відображає рівень підготовки користувача навчальної програмної системи. Отриманий вектор відображає рівень та здібностей знань користувача, історію помилок, а також траєкторію зміни його навичок.

Модуль визначення поточного стану користувача на основі біометричних даних формує вектор поточного стану користувача, що включає варіабельність та частоту серцевих скорочень й поточне когнітивне навантаження з відповідною міткою часу.

Модуль диспетчеризації навчальних модулів поєднує інформацію із векторів профілю та поточного стану користувача і на основі об'єднаного вектору підбирає навчальний модуль. При виборі модуля використовується

функція корисності, яка враховує рівень підготовки користувача і можливість когнітивного перенавантаження.

Модуль навчального контенту деталізує вибраний модулем диспетчеризації навчальний сценарій та передає цей сценарій на клієнтський рівень.

Модуль моніторингу нефункціональних характеристик записує часові мітки етапів процесу персоніфікованої адаптації, обчислює фактичні значення затримок, виявляє порушення заданих порогів і передає цю інформацію на рівень мікросервісів при завершенні сесії навчання.

Мікросервісний рівень виконує функції аналітики та верифікації, які не пов'язані із безпосередньою взаємодією з користувачем, проте забезпечують довгострокову адаптивну роботу програмної навчальної системи.

Даний рівень містить модулі $S = \{s_1, s_2, s_3, s_4\}$: s_1 – модуль верифікації; s_2 – модуль аналітики; s_3 – модуль генерації персоніфікованих сценаріїв навчання; s_4 – модуль даних системи навчання.

Модуль верифікації записує журнали подій (логи) проведених навчальних сесій і використовує ці логи для верифікації процесу персоніфікованої адаптації програмної системи з використанням технології process mining.

Модуль аналітики виконує агрегацію показників користувачів, зокрема динаміки рівня підготовки, розподілу станів здобувачів тощо. На основі агрегованих показників даний модуль формує звіти щодо успішності навчання для адміністрації автошколи.

Модуль генерації персоніфікованих сценаріїв навчання формує сценарій наступної сесії для користувача на основі історії його попередніх сесій.

Модуль даних системи навчання зберігає загальні дані про користувачів та процес навчання (профілі користувачів, завдання, журнали подій).

Розроблена гібридна архітектура адаптивної програмної навчальної системи представлена на рис. 2.2.

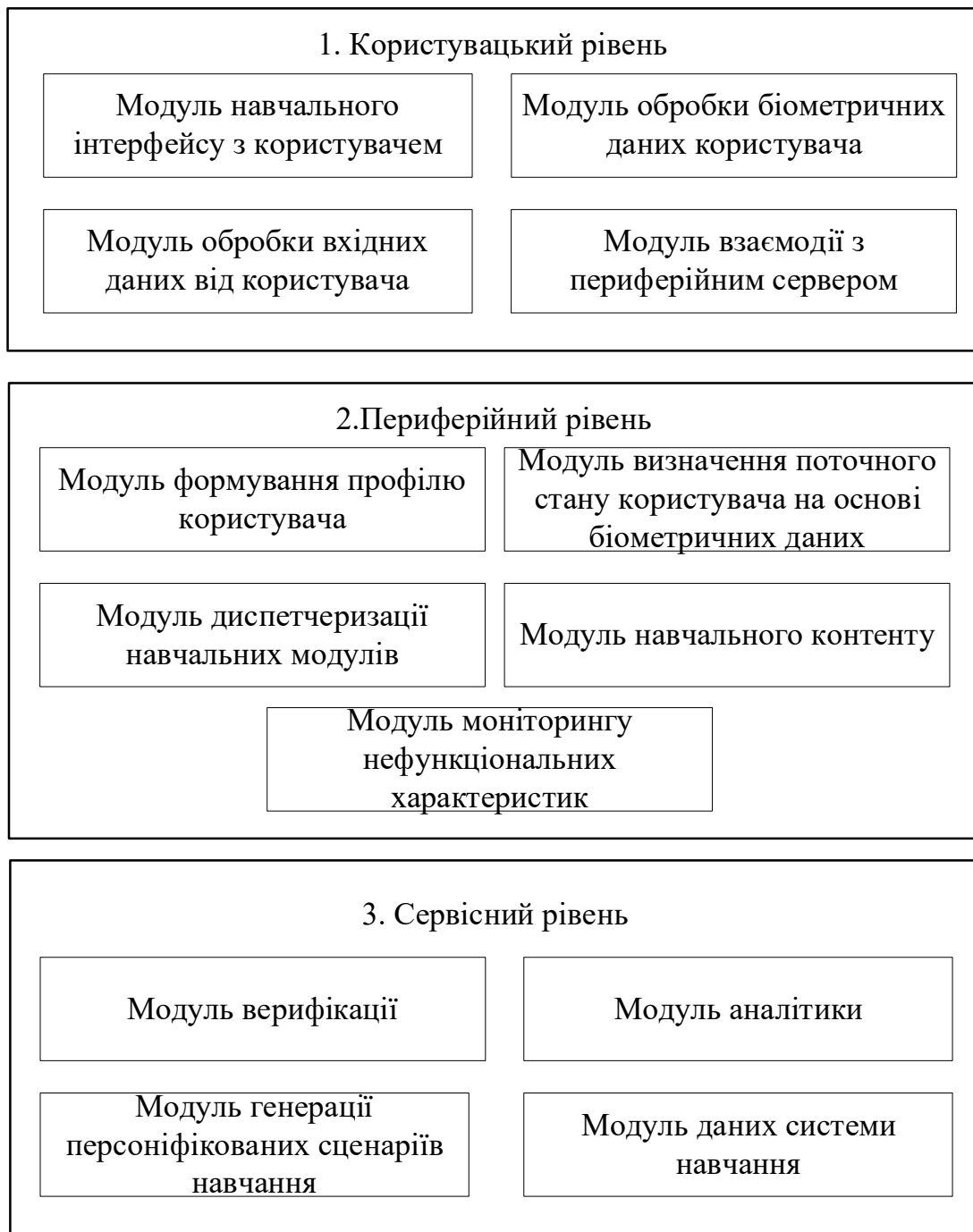


Рисунок 2.2 – Гібридна архітектура навчальної програмної системи

В запропонованій гібридній архітектурі реалізовано два потоки даних: синхронний та асинхронний.

Синхронний потік забезпечує процес персоналізованого адаптивного навчання з урахуванням поточного стану користувача і має задовольняти обмеження на затримку передачі даних. Сумарна затримка в даному потоці

складається із затримки на користувачькому та периферійному рівнях, а також затримки в комп'ютерній мережі при передачі даних між цими рівнями.

Асинхронний потік підтримує удосконалення процесу навчання і реалізує передачу даних із периферійного на мікросервісний рівень, де дані аналізуються і верифікуються. результати верифікації служать обґрунтуванням для подальшого удосконалення процесу навчання.

Потоки даних між рівнями гібридної програмної архітектури представлено на рис. 2.3.

Розглянемо ключові особливості розробленої гібридної архітектури. Дана архітектура поєднує декілька базових патернів взаємодії між модулями, причому кожен із патернів реалізує обмін даними на окремому рівні або між рівнями розробленої архітектури. При розробці гібридної архітектури враховано такі переваги та недоліки базових архітектур.

Монолітна архітектура забезпечує локальну обробку з низькою латентністю, що підходить для критичного шляху. Проте програмну систему з монолітною архітектурою потрібно розгорнути на кожному комп'ютері, тобто вона погана масштабується і тому є достатньо дорогою.

Подієво-орієнтований підхід призначений для масштабованих програмних систем. Проте такий підхід має великі затримки на критичному шляху обробки, якщо виконується асинхронна обробка запитів, бо запити ставляться в чергу.

Клієнт-серверна архітектура, що реалізована у глобальній мережі, забезпечує зниження витрат за рахунок тонких клієнтів. Однак у такій архітектурі затримки залежать від взаємодії з хмарою і тому затримки можуть бути значними.

Архітектура з периферійними серверними обчисленнями забезпечує компроміс між затримками та вартістю, однак збільшення кількості одночасних користувачів обмежено можливостями периферійного сервера.

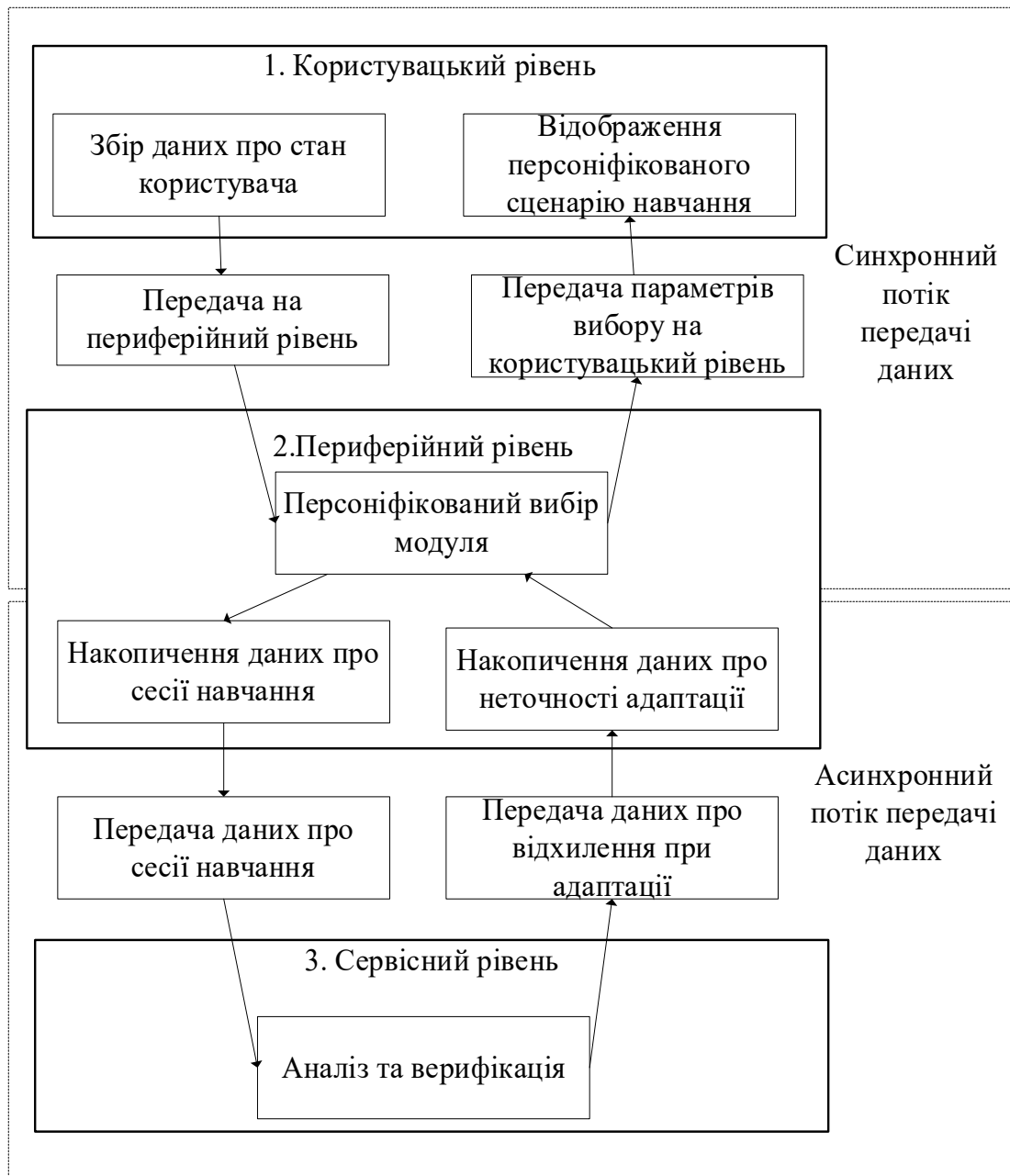


Рисунок 2.3 – Потіки даних в гібридній архітектурі програмної навчальної системи

Мікросервісна архітектура легко масштабується, але розгортається зазвичай у хмарі. Винесення частини логіки роботи до хмари збільшує затримки і суперечить вимогам до конфіденційності персоніфікованих біометричних даних.

Властивості базових архітектурних підходів, що були використані при розробці гібридної архітектури, наведено в табл. 2.2.

Таблиця 2.2 – Використання можливостей базових архітектурних підходів у розробленій гібридній програмній архітектурі

Базовий підхід	Використаний патерн	Переваги гібридної архітектури
Монолітна архітектура	Обробка в межах одного обчислювального процесу на периферійному рівні	Мінімальна внутрішня затримка між модулями на периферійному сервері
Клієнт-серверний підхід	Використання «тонкого клієнта» на навчальному місці користувача	Зниження вартості обладнання на клієнтському рівні
Архітектура, орієнтована на події	Асинхронний обмін подіями між периферійним і мікросервісним рівнями	Винесення ресурсоємних операцій за межі алгоритму адаптації навчання для користувача, тобто за границі критичного шляху обробки
Мікросервісна архітектура	Незалежне масштабування сервісів, наприклад через Kubernetes	Масштабування сервісів верифікації, аналітики та збереження даних без впливу на процес адаптації навчання.

Розроблена гібридна архітектура використовує можливості монолітного підходу на периферійному рівні, що забезпечує низьку затримку при спільному виконанні модулів d_1 – формування профілю користувача; d_2 – визначення поточного стану користувача на основі біометричних даних; d_3 – диспетчеризації навчальних модулів. Перевага клієнт-серверного підходу полягає в тому, що можна на навчальному місці користувача встановити недороге обладнання тонкого клієнту.

Подієвий підхід реалізовано між периферійним та мікросервісним рівнями гібридної архітектури через асинхронний обмін подіями (з використання, наприклад, Apache Kafka), що дозволяє винести ресурсоємні операції навчальної програмної системи за межі критичного шляху обробки біометричних даних користувача, де важливо забезпечити мінімальну затримку. Можливості масштабування мікросервісної архітектури забезпечують розширення сервісів

верифікації, аналітики та збереження даних про процес навчання без втручання в роботу алгоритму адаптації.

Таким чином, розроблена трирівнева модель гібридної програмної архітектури з периферійними обчисленнями системи навчання в автошколі поєднує можливості монолітної, клієнт-серверної, подієвої та мікросервісної архітектур, використовуючи їх переваги на клієнтському, периферійному та мікросервісному рівнях з урахуванням нефункціональних обмежень до персоніфікованого адаптивного навчання в автошколі.

Висновки до розділу 2

1. Розроблено адаптивну модель дорожньої ситуації для підтримки теоретичного навчання водіїв. Модель містить опис дорожньої ситуації засобами дескриптивної логіки, а також оцінку рівня складності цих ситуацій з використанням показників кількості об'єктів у складі дорожньої ситуації, часу на прийняття рішення щодо дії у дорожній ситуації, кількості варіантів можливих дій, які курсант автошколи може виконати у поточній дорожній ситуації. Розроблена модель створює умови для персоніфікованої адаптації процесу підготовки водіїв з урахуванням рівня їх знань та психофізіологічного стану.

2. Розроблено модель процесу обробки біометричних даних у програмних системах навчання водіїв в автошколах. Модель включає фази отримання біометричних даних, попередньої обробки даних, відбору первинних ознак, формування формального опису поточного стану користувача, передачу опису поточного стану користувача до модулю персоніфікованої адаптації процесу навчання. Модель створює умови для персоніфікації процесу навчання з урахуванням опису поточного психофізіологічного стану користувача, отриманого на основі його біометричних даних в режимі м'якого реального часу.

3. Вперше запропоновано модель гібридної програмної архітектури з периферійними обчисленнями системи навчання в автошколі. Модель містить

клієнтський, периферійний та мікросервісний рівні. Клієнтський рівень архітектури призначений для візуалізації дорожнього середовища. Периферійний рівень призначений для визначення психоемоційного стану користувача шляхом обробки його біометричних даних та подальшої персоніфікованої адаптації навчальних завдань. Мікросервісний рівень виконує верифікацію процесу персоніфікованої адаптації а також аналіз та узагальнення результатів процесу навчання в автошколі. Модель забезпечує можливість персоніфікованої адаптації процесу навчання у програмних системах автошкіл з урахуванням не лише рівня знань, а й поточного рівня когнітивного навантаження та відповідного психоемоційного стану користувача.

3 МЕТОДИ ПОБУДОВИ ПЕРСОНІФІКОВАНОГО АДАПТИВНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ НАВЧАННЯ ВОДІЇВ

3.1 Розробка методу формування вектору профілю користувача навчальної програмної системи

Розробка програмних систем для персоніфікованого адаптивного навчання користувачів із різними когнітивними здібностями, рівнем освіти та можливостями опанування навчального матеріалу є актуальною задачею у сфері інженерії програмного забезпечення, оскільки традиційні архітектурні підходи базуються на використанні статичного опису користувача. Статичний опис не враховує індивідуальні особливості й можливість удосконалення компетенцій при навчанні та не забезпечує адаптацію навчального процесу у реальному або близькому до реального часі. Даний недолік може привести до надання початківцям складних навчальних сценаріїв із надмірною функціональністю, або ж недостатньо складних модулів для просунутих користувачів. Внаслідок цього в подальшому відбувається некоректний підбір навчальних сценаріїв, оскільки вони не враховують індивідуальні патерни помилок користувачів при проходженні попередніх навчальних модулів. Тому в межах запропонованого в розділі 2 гібридного архітектурного рішення необхідно подолати такі недоліки традиційних архітектурних підходів в адаптивних програмних навчальних системах щодо побудови профілю стану користувача:

- обмежені можливості математичного забезпечення програмних компонент щодо автоматизованої оцінки рівня підготовки користувачів, що приводить до поділу користувачів за кількістю виконаних завдань без урахування складності цих завдань, а також патернів помилок;

- недостатня інтеграція можливостей психолого-педагогічних моделей у програмні архітектури систем навчання, що ускладнює розробку адаптивних навчальних систем, їх тестування та валідацію;

– відсутність неевристичних підходів до перетворення психометричних оцінок користувачів у дискретні рівні складності навчальних сценаріїв (модулів), що обмежує можливості адаптивної диспетчеризації навчальних компонент.

Формування вектору профілю користувача у програмних системах навчання безпосередньо впливає на вирішення наступних задач програмної інженерії:

- розробка архітектурних патернів для адаптивних систем;
- моделювання поведінки користувача як вхідного параметру для диспетчеризації навчальних програмних модулів;
- формалізація вимог до програмної компоненти оцінки стану користувача з урахуванням специфікації інтерфейсів та протоколів взаємодії між модулями;
- верифікація коректності алгоритмів адаптації шляхом порівняння запланованої, а також фактичної поведінки програмної системи навчання.

У контексті навчальних програмних систем для підготовки водіїв в автошколах некоректний підбір складності навчального сценарію приводить до когнітивного перевантаження (або недовантаження) користувача та недостатньої підготовки до дій в реальних дорожніх ситуаціях. Тому розробка методу формування профілю користувача має створити умови для задоволення нефункціональних вимог щодо затримок при адаптації системи, точності вибору навчального модуля та підтримки навчання користувачів із різним рівнем підготовки.

Аналіз літературних джерел показав відсутність комплексних рішень для навчальних програмних систем, що інтегрують психометричні моделі оцінки рівня підготовки користувачів у програмні компоненти з такими вимогами:

- формалізовані специфікації вхідних (вихідних) даних;
- обмеження щодо алгоритмічної складності обчислень;
- можливість валідації оцінки рівня підготовки користувача навчальної програмної системи шляхом порівняння з експертними оцінками інструкторів.

В дослідженнях [148, 149] розроблено трикомпонентну модель та метод оцінки рівня підготовки учнів на основі обробки даних щодо вирішення цими

учнями завдань в минулому. Проте задача імплементації даного підходу у програмні модулі у складі адаптивних навчальних систем із розподіленою архітектурою в даних роботах не розглядається. Роботи [150, 151] присвячені адаптації систем навчання водіїв з використанням біометричних даних. Проте вони не враховують попередню підготовку користувачів таких навчальних систем, що є важливим для персоніфікації базового рівня складності навчальних модулів до початку адаптації на основі біометричних даних.

Метою даного підрозділу є розробка методу формування профілю користувача у навчальній програмній системі на основі інтеграції попередніх психометричних оцінок рівня підготовки користувачів, аналізу результатів навчання та подальшої конвертації інформації щодо стану користувача у дискретні рівні складності для диспетчеризації навчальних модулів у гібридній програмній архітектурі.

Розроблений метод має забезпечити формування вектору профілю користувача як формалізованого програмного інтерфейсу для взаємодії з компонентами адаптивного підключення модулів навчальної системи (підрозділ 3.3) та контекстно-залежної верифікації з використанням методів process mining (підрозділ 3.4), що відповідає принципам модульності, слабкої зв'язності між компонентами та сильного зчеплення всередині модулів при проектуванні програмних архітектур [152].

Оцінка стану й наступне формування профілю користувача у програмній системі навчання базується на психометричній теорії оцінки результатів виконання завдання, що встановлює формальну залежність між здібностями й рівнем підготовки користувача, параметрами завдання та ймовірністю його правильного виконання користувачем системи. Психометрична теорія інтегрує три класичні підходи до оцінки здібностей учнів: тести фіксованої довжини; аналіз за шкалами (або шкальний аналіз); стандартні опитувальники.

Використання тестів фіксованої довжини передбачає, що всі учні отримують однаковий набір питань незалежно від їхнього рівня підготовки та знань. Наприклад, у стандартизованому тесті з 50 питань учень-початківець і

учень з високим рівнем знань виконують абсолютно ідентичні завдання. Основний недолік такого підходу полягає у неефективному використанні часу: слабкі учні витрачають багато часу на занадто складні питання, які не відповідають їхньому рівню розуміння, тоді як сильні учні змушені виконувати прості завдання, які не несуть для них діагностичної цінності [153]. Такий недиференційований підхід призводить до того, що тест фіксованої довжини добре працює лише для учнів середнього рівня підготовки.

Аналіз за шкалами ґрунтується на припущенні, що завдання можна впорядкувати від простих до складних, і успішне виконання складнішого завдання автоматично означає здатність виконати всі простіші. Такий підхід часто використовується у поетапному навчанні, де кожен рівень складності будується на попередньому. Проте головним обмеженням даного підходу є жорстка ієрархічна структура тестів, яка не враховує комплексність навичок учнів [153]. Наприклад, учень може добре виконувати практичні завдання високого рівня, але мати прогалини в теоретичних знаннях середнього рівня. Аналіз за шкалами також не передбачає групування завдань за «роздільною здатністю», тобто такі завдання не дають можливість відрізнити учнів з близькими рівнями навичок [154].

Стандартні опитувальники зазвичай містять серію питань із заздалегідь визначеними варіантами відповідей (множинний вибір, шкала Лайкерта тощо), які оцінюють знання або навички учня на основі його звіту за визначеними критеріями. Такі інструменти легко використовувати та стандартизувати, що обумовлює їх широке використання для масових оцінок. Однак ключовий недолік даного підходу полягає у відсутності адаптивності: опитувальник не змінює складність або спрямованість питань в залежності від попередніх відповідей учня [156]. Така властивість означає, що для досягнення оцінки прийнятної точності в опитувальник необхідно включати велику кількість питань. Збільшення набору питань суттєво подовжує час тестування і зменшує мотивацію учнів. Крім того, стандартні опитувальники часто не враховують додаткові фактори, які виникають у процесі навчання, наприклад когнітивне

перенавантаження, поточний емоційний стан учня тощо. Ці фактори можуть суттєво впливати на результати опитування.

Наведені обмеження класичних методів свідчать необхідність використання адаптивних систем оцінювання. Останні мають оперативно коригувати складність і призначення завдань на основі поточного рівня учня, забезпечуючи водночас підвищення якості навчання і зменшення витрат часу [154, 156].

Розроблений метод використовує концепцію комп'ютеризованого адаптивного тестування (Computerized Adaptive Testing, CAT), що поєднує теорію ймовірностей і адаптивність [154, 157]. CAT використовує базові ідеї тестування пацієнта у лікаря: якщо пацієнт легко виконав базові тести руху, то лікар одразу переходить до більш складних вправ, достроково завершуючи перевірку базових навичок.

Концепція комп'ютеризованого адаптивного тестування полягає в тому, що вибір наступного питання має залежати від того, чи правильно учень відповів на попереднє. Якщо відповідь учня правильна, то система тестування пропонує вирішити складніше завдання, в іншому випадку надає простіший тест. Такий підхід дає можливість більш точно оцінити здібності учня з використанням суттєво меншої кількості питань у порівнянні з тестами фіксованої довжини. Дана концепція реалізується в рамках теорії відгуку на завдання (Item Response Theory, IRT). Остання трактує ймовірність успішного проходження поточного навчального епізоду як функцію від здібностей учня та параметрів кожного завдання, що визначає взаємодію між характеристиками особи та характеристиками завдання [154]. Даний підхід дозволяє передбачити ймовірність успіху в конкретній навчальній ситуації.

Функція адаптивної взаємодії спирається на модель [153, яка трансформує безперервну вхідну змінну у значення ймовірності. Згідно цієї моделі, ймовірність $P(\text{result}(a_i, b_j) = \text{true})$ правильної відповіді учня зі здібностями b_j на завдання зі складністю a_i визначається різницею між його здібностями та

складністю завдання. Латентна здібність користувача b_j зазвичай представляється як безрозмірна величина у діапазоні $[-3; 3]$, де нульове значення відповідає середньому рівню підготовки. Складність завдання a_i має таку саму шкалу, що й здібності. Тобто складність $a_i = 1$ означає, що для розв'язання завдання учень повинен мати здібності $b_j = 1$. Якщо здібності учня значно перевищують складність завдання, то ймовірність коректної відповіді наближається до 1, а в протилежному випадку – до 0. Тобто ця залежність є нелінійною і формалізується за допомогою логістичної функції або сигмоїди:

$$P(\text{result}(a_i, b_j) = \text{true}) = \frac{1}{1 + e^{-(a_i - b_j)}}. \quad (3.1)$$

Однак модель оцінки ймовірності правильної відповіді (3.1) не враховує, що деякі сценарії завдань при навчанні в автошколі краще розділяють слабких та сильних учнів, ніж інші. Наприклад, при навчанні водінню маневр «паралельне паркування» чітко розділяє можливості новачків і досвідчених водіїв, а завдання «рух по прямій дорозі» виконують успішно майже всі курсанти автошколи, незалежно від рівня їх майстерності.

Для усунення цього недоліку в [155] запропоновано ввести параметр розрізнення c_i , що визначає чутливість завдання до рівня здібностей через крутизну логістичної кривої, а також параметр вгадування d_i , що задає мінімально можливе значення ймовірності. Високе значення параметру дискримінації c_i означає, що навіть невелика зміна в здібностях учня суттєво впливає на ймовірність успіху (крива сигмоїди стає більш крутою), а низьке c_i свідчить про “розмитість” завдання, яке слабо розрізняє учнів різних рівнів.

На практиці зазвичай використовується діапазон значень параметра дискримінації в діапазоні $[0,5; 2,5]$, оскільки значення $c_i > 1,5$ дає можливість ефективно відокремити підготовлених користувачів навчальної системи від учнів з низьким рівнем підготовки.

При використанні N_{sc} навчальних модулів, що реалізують різні сценарії представлення знань, $d_i = \frac{1}{N_{sc}}$, тобто даний параметр відповідає простому вгадуванню результату учнем. Тоді модель [155] для задачі адаптивного підбору навчального модуля для користувача приймає вигляд:

$$P\left(result(a_i, b_j)=true \middle| c_i, \frac{1}{N_{sc}}\right) = \frac{1}{N_{sc}} + \left(1 - \frac{1}{N_{sc}}\right) \frac{1}{1 + e^{-c_i(a_i - b_j)}}. \quad (3.2)$$

Ця модель дозволяє на кожному кроці навчання обрати найбільш інформативне завдання, тобто завдання, відповідь на яке є найбільш невизначеним для учня, тобто $P\left(result(a_i, b_j)=true \middle| c_i, \frac{1}{N_{sc}}\right) = 0,5$.

Ймовірність правильної відповіді складається з двох компонент – базової ймовірності вгадування $\frac{1}{N_{sc}}$ та додаткової ймовірності, яка залежить від різниці між латентними здібностями користувача й складністю завдання.

Латентна здібність b_j користувача оцінюється методом максимальної правдоподібності (Maximum Likelihood Estimation, MLE) на основі його відповідей на множину завдань $Q = \{q_i\}$. Функція правдоподібності $L(a_i|Q)$ для задачі адаптивного підбору навчального модуля для користувача приймає вигляд:

$$\begin{aligned} L(a_i|Q) &= \\ &= \prod_i P\left(result(a_i, b_j)=true \middle| c_i, \frac{1}{N_{sc}}\right)^{q_i} \cdot [1 - P\left(result(a_i, b_j)=true \middle| c_i, \frac{1}{N_{sc}}\right)]^{1-q_i}. \end{aligned} \quad (3.3)$$

Функція правдоподібності оцінює, наскільки ймовірним є отриманий набір відповідей Q для користувача з латентною здібністю b_j . Наприклад, розглянемо ситуацію, коли користувач правильно відповів на складні завдання та неправильно на прості, тобто ця комбінація відповідей нетипова для любого значення латентних здібностей b_j . В такому випадку значення $L(a_i|Q)$ буде низьким для всіх значень b_j , що відображає нестабільність відповідей або випадковість.

У випадку, якщо користувач надає правильні відповіді на завдання зі складністю a_i , то $L(a_i|Q)$ досягає максимуму при певному значенні латентних здібностей. Тоді оцінка когнітивних здібностей користувача b_j^* визначається як значення аргументу b_j , що максимізує функцію правдоподібності $L(a_i|Q)$:

$$b_j^* = \underset{b_j}{\operatorname{argmax}} L(a_i|Q). \quad (3.4)$$

На основі поточної оцінки когнітивної здібності (3.4) може бути здійснений динамічний вибір наступного завдання для користувача. Такий вибір здійснюється з використанням алгоритму САТ.

Якщо користувач відповідає правильно, то він отримує більш складне наступне завдання з більшим значенням a_i , якщо неправильно – отримує завдання з меншим значенням a_i . Такий алгоритм дає можливість швидше оцінити рівень підготовки здобувача [157, 158].

Вибір наступного завдання здійснюється за оцінкою інформації Фішера $I(a_i)$, яка кількісно описує, скільки інформації про завдання містить відповідь користувача.

Дана оцінка використовує першу похідну від ймовірності правильної відповіді. Для спрощення запису позначимо $P\left(\operatorname{result}(a_i, b_j)=\text{true} \middle| c_i, \frac{1}{N_{Sc}}\right)$ як P_{ij} .

Тоді оцінка Фішера приймає вигляд:

$$I(a_i) = b_j^2 \cdot \frac{(P_{ij}')^2}{P_{ij} \cdot (1 - P_{ij})}. \quad (3.5)$$

Оцінка (3.5) досягає максимуму для завдань, складність яких a_i близька до поточної оцінки b_j користувача. Занадто легкі та занадто складні завдання містять мало інформації, оскільки користувач майже напевно відповість правильно на перші і неправильно на другі завдання, що не дає можливість звужити діапазон значень a_i . Тому вибір номера k наступного завдання здійснюється на множині невиконаних завдань, тобто завдань для яких виконується умова $i < k$, за критерієм:

$$k = \operatorname{argmax}_{i| i < k} I(a_i). \quad (3.6)$$

Вибір завдань (3.6) може бути достатньо складним для класів практичних задач навчанні водінню авто, управлінню складними технічними системами тощо внаслідок великої кількості різнопланових підзадач. Для спрощення вибору завдання розділяються на рівні на основі порогових значень здібностей користувача. Такий підхід спрощує диспетчеризації навчальних модулів у програмній системі.

Рівні складності L_m мають вигляд:

$$L_m = \begin{cases} L1, & \text{iff } b_j^* < -1,0, \\ L2, & \text{iff } -1,0 \leq b_j^* < -0,3, \\ L3, & \text{iff } -0,3 \leq b_j^* < +0,5, \\ L4, & \text{iff } +0,5 \leq b_j^* < +1,2, \\ L5, & \text{iff } b_j^* \geq +1,2. \end{cases} \quad (3.7)$$

Відповідно до (3.7), користувачі з низькими $b_j^* < -1,0$ потребують базових навчальних модулів рівня L1 без обмежень часу, наприклад ситуації водіння на порожньому паркувальному майданчику, на вулиці, де є ще 1–2 автомобілі. Користувачі з середніми $-0,3 \leq b_j^* < +0,5$ отримують можливість вивчити міські сценарії L3 (наприклад, 15–25 автомобілів, дорожня ситуація на перехресті). Користувачі з високими $b_j^* \geq +1,2$ отримують експертні навчальні модулі L5 (наприклад, 50 і більше автомобілів на дорозі, виникнення екстремальних ситуацій).

Такий підхід дає можливість організувати доступ до модулів у зоні найближчого розвитку навичок та вмінь користувача, дещо складніших за його поточний рівень користувача. Використання модулів з незначним підвищенням складності завдань стимулює розвиток навичок без когнітивного перевантаження [159].

Окрім оцінки рівня латентних здібностей, для формування повного вектору стану умінь користувача необхідно врахувати типові помилки користувача U_{error} та траєкторію зміни його навичок з часом U_{trend} . Ці показники характеризують рівень засвоєння навчальних модулів користувачем. Результируючий вектор профілю помилок має вигляд:

$$U = \langle U_{\text{error}}, U_{\text{trend}} \rangle. \quad (3.8)$$

Профіль помилок $U_{\text{error}} = \{U_{\text{error}}^{(l)}\}$ містить частоти помилок по рівнях важливості, від стратегічних до ситуативних помилок. Рівні важливості l включають такі типи помилок:

– стратегічні, які відображають порушення загальних закономірностей при вирішенні задач; наприклад при опануванні теоретичних знань при навчанні водінню стратегічні помилки включають невірний вибір траєкторії, необґрунтоване рішення про обгін тощо;

– тактичні, які виникають при вирішенні конкретних практичних завдань; наприклад при навчанні водінню до таких помилок відносяться занадто різке гальмування, невідповідна швидкість на повороті;

– ситуативні, що виникають у відповідь на непередбачені події; наприклад для майбутніх водіїв ситуативними помилками є затримка реакції на пішохода, неправильна оцінка дистанції.

Для кожної категорії помилок обчислюється їх частка як відношення до загальної кількості помилок даного рівня:

$$U_{\text{error}}^{(l)} = \frac{\text{кількість помилок рівня } l}{\text{загальна кількість дій рівня } l}. \quad (3.9)$$

Траєкторія зміни навичок U_{trend} моделює послідовність зміни b_j з часом для прогнозування зони найближчого розвитку користувача. Якщо користувач виконав тестові завдання у дискретні моменти часу $1, 2, \dots, t, \dots, T$, то послідовність оцінок його здібностей має вигляд:

$$B = \langle b_{1j}, b_{2j}, \dots, b_{tj}, \dots, b_{Tj} \rangle. \quad (3.10)$$

Така траєкторія у найпростішому випадку апроксимується лінійною регресією. Значення когнітивних здібностей користувача у майбутні моменти часу $b_j^*(t)$ має вигляд:

$$b_j^*(t) = \theta_0 + \theta_1 t, \quad (3.11)$$

де θ_0 – зміщення;

θ_1 – коефіцієнт нахилу, що характеризує швидкість розвитку навичок та умінь користувача.

Розбиття значень θ_1 на порогові інтервали дає можливість визначити правил вибору навчальних модулів для користувача. Наприклад, якщо $\theta_1 > 0,2$, то користувач швидко навчається і програмна система навчання може підвищити складність завдань.

Оцінка рівня підготовки і латентних здібностей користувача має вигляд вектора V_{user}^1 , який включає рівні складності модулів для користувача, поточну оцінку здібностей користувача, частоту помилок та траєкторію зміни рівня підготовки користувача навчальної програмної системи:

$$V_{\text{user}}^1 = \langle L_m, b_j^*, U_{\text{error}}, U_{\text{trend}} \rangle. \quad (3.12)$$

Метод формування вектору профілю користувача у навчальній програмній системі містить такі етапи [3].

Етап 1. Побудова вхідного набору даних.

На даному етапі виконується агрегація результатів виконання завдань користувачем, що містять інформацію про правильність відповідей, час виконання, тип завдання тощо. При виконанні методу формується набір даних $E = \{e_i\}$.

Етап 2. Оцінка рівня підготовки користувача a_i .

При виконанні даного етапу оцінка рівня підготовки користувача розраховується на основі функції правдоподібності (3.3) як значення аргументу, що максимізує цю функцію (3.4).

Етап 3. Уточнення рівня підготовки користувача.

На даному етапі неодноразово здійснюється вибір тестового завдання з використанням оцінки Фішера (3.5). Тестові завдання вибираються за формулою (3.6). Підбирається завдання, складність якого близька до поточної оцінки рівня підготовки користувача навчальної системи. Даний етап виконується повторно до досягнення заданої точності оцінки латентних здібностей користувача.

Етап 4. Перетворення отриманої оцінки рівня підготовки користувача у рівні складності завдань для користувача.

Перетворення виконується з використанням порогових значень оцінки латентних здібностей згідно (3.7), що дає можливість узгодити рівень підготовки користувача і складність навчальних модулів для алгоритму диспетчеризації (підрозділ 3.3).

Етап 5. Формування профілю типових помилок.

При виконанні етапу використовуються параметри завдань $E = \{e_i\}$, мітки категорій помилок, а також оцінки користувача у попередні моменти часу. Завдання класифікуються за категоріями помилок і обчислюється частоти помилок у кожній категорії відповідно до (3.9). З використанням лінійної регресії (3.11) формується траєкторія змін помилок. Результатом етапу є вектор профілю помилок (3.8), який містить стратегічні тактичні та ситуаційні помилки, що дає можливість визначити патерни прискореного або уповільненого навчання.

Етап 6. Формування вектору профілю користувача навчальної програмної системи.

При формуванні вектору профілю користувача об'єднуються результати попередніх етапів. Отриманий вектор (3.12) враховує рівень складності завдання для користувача (результат етапу 4), рівень підготовки користувача (результат етапу 3), профіль помилок (результат етапу 5). Вектор може бути представлений у форматі JSON для подальшого використання при диспетчеризації навчальних модулів.

Розроблений метод формування профілю користувача у навчальній програмній системі має ряд переваг порівняно з існуючими підходами до оцінки користувачів. Метод виконує комплексну оцінку індивідуальних характеристик кожного завдання з використанням параметрів розрізнення, складності та вгадування, що дає можливість прискорити навчання з використанням модулів складності відповідно до рівня підготовки користувача. Метод визначає дискретні рівні складності завдань на основі цієї оцінки, що дає можливість уникнути ручного налаштування порогових значень у процесі розробки

програмного забезпечення. Формування профілю помилок та траєкторії розвитку навичок користувача дає можливість підібрати навчальні модулі відповідно до сильних та слабих сторін користувача при диспетчеризації модулів. Отриманий при виконанні методу вектор профілю користувача забезпечує специфікацію інтерфейсу при взаємодії з компонентами диспетчеризації модулів (підрозділ 3.3) та верифікації (підрозділ 3.4), що відповідає принципам модульності, слабого зв'язку та високої зв'язності при проектуванні програмних архітектур.

Розроблений метод використовує загальні психометричні принципи оцінки користувачів, інваріантних до предметної галузі, що дає можливість створювати бібліотеки компонент попередньої оцінки стану користувача для навчальних систем з різних предметних галузей, знижуючи вартість розробки через повторне використання коду та масштабування архітектурних рішень.

3.2 Розробка методу побудови вектору поточного стану користувача в навчальній програмній системі

Даний підрозділ присвячений розробці методу формування вектору стану користувача, що забезпечує персоніфікацію процесу навчання у адаптивних навчальних програмних системах і доповнює профіль користувача, отриманий з використанням розробленого у підрозділі 3.1 методу. Доповнення профілю користувача враховує поточний його психофізіологічний під час взаємодії з навчальною програмною системою. Такий стан може включати когнітивне перевантаження, стрес, втому, зниження концентрації уваги. Ці фактори можуть суттєво знизити ефективність навчання навіть при підбраному відповідно до профілю користувача базовому рівні складності навчальних модулів [160, 161]. Опис поточного стану користувача в навчальній програмній системі забезпечує персоніфіковану адаптацію навчального матеріалу з урахуванням як рівня знань, так і поточного психоемоційного стану користувача.

Існуючі програмні рішення для моніторингу стану користувача в навчальних системах мають ряд обмежень щодо персоніфікації навчальних сценаріїв [161-163]. Ці обмеження пов'язані із складністю інтеграції у навчальні програмні системи методів біометричного моніторингу та відсутністю формальних програмних інтерфейсів для обробки поточного стану користувача в режимі реального часу.

Складність інтеграції методів біометричного моніторингу виникає тому, що такі методи мають бути реалізовані у окремих програмних компонентах із визначеними інтерфейсами для біометричних даних, що передаються із носимих сенсорів, з можливостями фільтрації «сирих» біометричних даних та їх класифікації. Дане обмеження на практиці приводить до використання спрощеного процесу обробки біометричних даних на основі евристичних правил зі статичними пороговими значеннями. Наприклад, частота серцевих скорочень вище заданого порогового значення цими правилами може бути інтерпретована як індикатор стресового стану користувача програмної системи під час опанування поточного навчального модуля. Однак така підвищена частота роботи серця може бути, наприклад, наслідком не психічного, а попереднього фізичного навантаження.

Відсутність визначених програмних інтерфейсів для оцінки стану користувача програмної системи в режимі реального часу обмежує можливості персоніфікованої адаптації навчальних сценаріїв та верифікації процесу адаптивного навчання. Тобто відсутність інтерфейсів обмежує можливості реалізації адаптивної програмної архітектури із персоніфікацією на основі оцінки як профілю, так і поточного стану користувача.

Таким чином, оцінка стану користувача в режимі реального часу потребує розробки програмних інтерфейсів для інтеграції оброблених даних щодо поточного стану користувача із зовнішніх носимих сенсорів для подальшої персоніфікованої адаптації навчальних модулів у процесі навчання.

Аналіз літературних джерел щодо розробки адаптивних програмних навчальних систем показав, що в сучасних дослідженнях недостатньо уваги

приділяється інтеграції результатів біометричного моніторингу (зокрема, на основі аналізу частоти та варіабельності серцевого ритму) з використанням специфікацій програмного інтерфейсу на основі вектору стану користувача [161, 163]. Дослідження можливостей використання варіабельності серцевого ритму як індикатора когнітивного навантаження в освітніх системах підтверджують кореляцію серцевого ритму з когнітивним навантаженням [160, 163]. В дослідженні [160] персоніфікація виконується на основі евристичних правил із статичними порогами, що враховують частоту серцевих скорочень, проте не враховують розвиток навичок користувача, що обмежує точність класифікації. Дослідження [162] представляє метод адаптації на основі електроенцефалографічного моніторингу лобного співвідношення тета/альфа хвиль для класифікації когнітивного навантаження. Метод має високу точність визначення стану користувача, однак використовує дорогий багатоканальний шолом для електроенцефалографії та характеризується суттєвою затримкою при зборі й обробці сигналу.

Таким чином, вирішення задачі оцінки поточного стану користувача навчальної програмної системи в режимі реального часу на основі носимого сенсору доступної вартості потребує розробки методу побудови вектору поточного стану для подальшого з створення міжмодульних інтерфейсів у програмній системі.

Розроблений метод оцінки поточного стану користувача в навчальній програмній системі використовує аналіз варіабельності серцевого ритму (Heart Rate Variability, HRV). Цей фізіологічний показник відображає варіацію часових інтервалів між послідовними ударами серця. Останні реєструються як інтервали R-R на електрокардіограмі або ж як інтервали між піками пульсової хвилі при вимірюванні сенсорами носимих пристроїв [167].

На відміну від простого вимірювання частоти серцевих скорочень (Heart Rate, HR), що надає лише середнє значення ударів за хвилину, варіабельність серцевого ритму характеризує невеликі відхилення між ударами пульсу. Ці відхилення контролюються автономною нервовою системою, відповідальною за

реакцію на стрес у вигляді підвищення серцевої активності та мобілізації організму при загрозі чи когнітивному навантаженні, а також за зниження активності серця у спокійному стані. Тому використання показника варіабельності серцевого ритму є умовою об'єктивної оцінки поточного психофізіологічного стану користувача навчальної системи [164,165]. Висока варіабельність HRV свідчить про спокійний розслаблений стан користувача, а низька варіабельність відображає когнітивне перевантаження та стрес, що дає можливість персоніфікувати рівень складності навчальних сценаріїв у програмній системі навчання [166].

Для оцінки варіабельності серцевого ритму традиційно використовується, різниця ΔRR інтервалів серцевих скорочень RR_n при n вимірі та RR_{n-1} при попередньому вимірі $n-1$:

$$\Delta RR = RR_n - RR_{n-1}. \quad (3.13)$$

Великі значення ΔRR свідчать про релаксацію, а малі значення показують стресовий стан людини. Інтервал серцевих скорочень RR_n визначається для послідовних n та $n-1$ ударів серця. Даний інтервал зазвичай вимірюється в мілісекундах між піковими точками пульсової хвилі.

На практиці вікно вимірювань пульсу доцільно встановити в 1 хвилину. Виходячи з розрахунку пульсу в спокійному стані 60 ударів на хвилину кількість вимірювань зазвичай становить $N = 60$.

Середньоквадратичне відхилення для різниці ΔRR між послідовними вимірами RR_n описує поточну варіабельність серцевих скорочень:

$$HRV = \sqrt{\frac{\sum_{n=2}^N \Delta RR^2}{N-1}}. \quad (3.14)$$

Якщо користувач навчальної програмної системи перебуває у спокійному стані, з низьким когнітивним навантаженням, то ΔRR суттєво змінюється внаслідок, наприклад, фізичних рухів. Якщо ж користувач знаходиться у стресовому стані, то інтервали RR_n змінюються несуттєво, тому різниця ΔRR буде мінімальною. Такі значення ΔRR свідчать про високе когнітивне навантаження і необхідність заміни навчального модуля. Таким чином, використання аналізу варіабельності серцевого ритму HRV, на відміну від частоти серцевих скорочень HR, дає можливість розрізнити фізичне навантаження від когнітивного навантаження, оскільки для фізичного навантаження характерна висока, а для когнітивного перевантаження – низька HRV.

Додатково до HRV (3.14) як показник поточного стану користувача навчальної системи може бути розраховано відношення частотних компонент варіабельності серцевого ритму. Співвідношення низькочастотних та високочастотних складових $HRV = \frac{HRV_{low}}{HRV_{high}}$ свідчить про стан людини. Значення співвідношення більше 2-х свідчить про когнітивне перевантаження, а значення менше 1 – про спокійний стан [166]. Для розрахунку даного відношення використовується швидке перетворення Фур'є. Результатом перетворення Фур'є є спектральна щільність потужності серцевого ритму, яка інтегрується окремо для низько частотної та високочастотної складової [164, 165].

В якості скалярного індикатора поточного стану користувача навчальної програмної системи розроблений метод використовує інтегральний показник стану серця H , який враховує нормалізовану варіабельність серцевих скорочень HRV_{norm} та нормалізовану частоту серцевих скорочень HR_{norm} :

$$H = \frac{HRV_{norm} + HR_{norm}}{2}. \quad (3.15)$$

Нормалізована частота серцевих скорочень HR_{norm} розраховується через співвідношення поточного значення HR_{curr} та значення $HR_{typical}$ у спокійному стані людини:

$$HR_{norm} = \frac{HR_{curr} - HR_{typical}}{HR_{typical}}. \quad (3.16)$$

Додатково в (3.15) може бути використана компонента $\frac{HRV_{low}}{HRV_{high}}$. Однак включення до складу H даного показника потребує обґрунтованого вибору вагових коефіцієнтів для всіх трьох компонент.

Для автоматизованої класифікації поточного стану користувача на три категорії когнітивного навантаження на основі часової послідовності ознак стану користувача використовується мережа довгої короткочасної пам'яті LSTM, яка призначена для виділення патернів у часі [7].

Крім ознак HRV_{norm} , HR_{norm} , H , для LSTM в якості елементів вхідних даних використовуються стандартне відхилення RR_{sdnn} на інтервалі спостереження із N вимірів, $\Delta RR_{threshold}$ – відсоток ΔRR , що перевищує порогове значення, середня частота серцевих скорочень HR_{mean} , коефіцієнт αHR лінійної регресії, що відображає тренд зміни серцевих скорочень (збільшення або зменшення).

Нормалізація HRV_{norm} , HR_{norm} дає можливість без додаткових вагових коефіцієнтів обробляти дані різних користувачів.

Агрегований показник стану серця H надає LSTM додаткову інформацію у порівнянні з можливими нелінійними комбінаціями HRV_{norm} , HR_{norm} .

Стандартне відхилення інтервалів RR_{sdnn} розраховується з урахуванням середнього значення інтервалів RR_{mean} та дає можливість виявити довгострокові тренди у послідовності серцевих скорочень:

$$RR_{sdnn} = \left(\frac{\sum_{n=1}^N (RR_n - RR_{mean})^2}{N} \right)^{\frac{1}{2}}. \quad (3.17)$$

Відсоток різниці між інтервалами серцевих скорочень $\Delta RR_{threshold}$, що перевищує порогове значення, використовується для виявлення переходів між станами низької, нормальної і підвищеної когнітивної активності.

Коефіцієнт αHR визначається при побудові лінійної регресії, задає динаміку зміни серцевих скорочень на поточному інтервалі часу, що дозволяє розрізнити перехід до нового стану когнітивної активності.

LSTM виявляє темпоральні патерни зміни наведених показників у часі: поступове наростання, пік, стабілізація. В рамках типового інтервалу вимірювань 60 секунд для LSTM в якості вхідних даних може бути подано 10-12 серій вхідних даних при вимірах протягом 6 і 5 секунд відповідно.

Результатом класифікації є поточний стан когнітивного навантаження користувача Cog_{user} , який може приймати одне з трьох значень {Високе навантаження; Норма; Низьке навантаження}.

Фінальний вектор поточного стану користувача V_{user}^2 включає нормовані значення HRV_{norm} , HR_{norm} , поточний стан когнітивного навантаження B_{user} та мітку часу $t_{current}$:

$$V_{user}^2 = \langle HRV_{norm}, HR_{norm}, Cog_{user}, t_{current} \rangle. \quad (3.18)$$

Вектор V_{user}^2 містить інформацію щодо поточного стану користувача у вигляді, призначеному для обробки компонентом диспетчеризації модулів в навчальній програмній системі. Кожна складова вектору містить інформацію про різні аспекти поточного стану, що дає можливість приймати багатокритеріальні рішення щодо диспетчеризації модулів навчальної системи. Наприклад, підвищення HR_{norm} , зниження HRV_{norm} , та значення Cog_{user} «Високе

навантаження» свідчать про перевищення когнітивного навантаження користувача та необхідність перемикавання на спрощений навчальний модуль.

Розроблений метод реалізує так послідовність етапів з побудови вектору стану користувача [3].

Етап 1. Збір біометричних даних.

На першому етапі ініціалізується з'єднання з сенсором серцевого ритму за допомогою бездротових протоколів низького енергоспоживання, передача інтервалів часу між серцевими скороченнями, синхронізація міток часу із системним годинником, формування потоку необроблених біометричних даних для подальшої обробки.

Етап 2. Фільтрація та нормалізація даних про поточний стан користувача.

Виконується медіанна фільтрація для частоти серцевих скорочень для видалення різких змін частоти серцевих скорочень. При фільтрації виявляються екстрасистולי на інтервалах між серцевими скороченнями, що відхиляються понад певний відсоток від медіани. Подальша нормалізація виконується відносно спокійного стану перед сесією навчання.

Етап 3. Розрахунок ознак варіабельності серцевого ритму.

На третьому етапі обчислюється середньоквадратичне відхилення варіабельності серцевого ритму з використанням виразу (3.14). Додатково може бути розраховано співвідношення низькочастотної та високочастотної складових серцевого ритму. Етап включає наступні кроки.

Крок 3.1. Розрахунок різниці ΔRR інтервалів серцевих скорочень.

Крок 3.2. Обчислення середньоквадратичного відхилення варіабельності серцевого ритму.

Крок 3.3. Нормалізація середньоквадратичного відхилення варіабельності серцевого ритму.

Етап 4. Розрахунок інтегрального показнику стану серця H .

Інтегральний показник стану серця H агрегує ознаки серцевого ритму у єдиний скалярний індикатор стану користувача згідно з формулою (3.15). Даний етап передбачає реалізацію таких кроків.

Крок 4.1. Обчислення нормалізованої частоти серцевих скорочень.

Крок 4.2. Розрахунок інтегрального показника H .

Етап 5. Класифікація когнітивного навантаження користувача навчальної програмної системи.

Класифікація виконується з використанням довгої короткочасної пам'яті LSTM та стандартного відхилення інтервалів RR_{sdnn} (3.17). Етап включає наступні кроки.

Крок 5.1. Обчислення стандартного відхилення інтервалів.

Крок 5.2. Класифікація когнітивного навантаження з використанням LSTM.

Етап 6. Формування вектору V_{user}^2 поточного стану користувача навчальної програмної системи.

Вектор містить (3.18) містить компоненти, що характеризують поточний стан користувача HR_{norm} , зміни стану HRV_{norm} , класифікацію поточного стану та мітку часу.

При практичному використанні даний вектор представляється у JSON – форматі і в такому форматі використовується у методах персоніфікованої адаптації процесу вибору навчальних модулів (підрозділ 3.3) та верифікації (підрозділ 3.4).

Розроблений метод онлайн-оцінки стану користувача у навчальній програмній системі має ряд переваг порівняно з існуючими підходами до моніторингу психофізіологічного стану користувачів.

Зокрема, поточний метод не обмежує користувача у процесі навчання і не потребує додаткової підготовки до вимірювань, тоді як методи точної оцінки когнітивного навантаження на базі електроенцефалографії використовують електроди, які вимагають складної процедури підготовки користувача, потребують для обслуговування кваліфікованих спеціалістів, що обмежує їх використання у навчальних програмних системах.

Розроблений метод представляє стан користувача у вигляді вектору поточного стану, який може бути реалізовано як програмний об'єкт із чітко

визначеними типами полів, зокрема у форматі JSON. Такий програмний об'єкт специфікує інтерфейс з модулем диспетчеризації, який забезпечує персоніфіковану адаптацію процесу підключення навчальних модулів програмної системи.

Розроблені в підрозділах 3.1 та 3.2 методи забезпечують двофазну оцінку стану користувача навчальної програмної системи на основі поєднання векторів профілю та поточного стану користувача, що дає можливість підібрати навчальні модулі в залежності як від рівня підготовки, так і поточного психофізичного стану курсанта автошколи, тим самим підвищивши ефективність засвоєння навчального матеріалу.

Існуючі підходи до класифікації когнітивного навантаження у адаптивних навчальних системах зазвичай базуються на евристичних порогових правилах з використанням біометричних показників, наприклад «якщо частота серцевих скорочень перевищує порогове значення», то користувач має когнітивне перенавантаження. Однак такі правила не враховують індивідуальні відмінності у частоті серцевих скорочень залежно від віку, рівня фізичної підготовки, статі та не дають можливість розрізнити фізичне навантаження. Зокрема, рухи людини підвищують частоту серцевих скорочень навіть без зміни когнітивного навантаження.

Розроблений метод виконує класифікацію за допомогою мережі LSTM, яка дає можливість врахувати такі темпоральні патерни, як поступове наростання стресового стану, пікові навантаження, стабілізація стану після перевантаження, і підвищити точність визначення стану перенавантаження користувача навчальної програмної системи.

Проте розроблений метод має ряд обмежень. Зокрема, точність обчислення нормалізованої частоти серцевих скорочень та нормалізованої варіабельності серцевого ритму на етапах 2 та 3 методу залежать від коректності вимірювання базових параметрів користувача у спокійному стані перед початком навчальної сесії. Також точність вимірювання частоти серцевих скорочень чутлива до рухів користувача. Зокрема, рухи руками можуть привести до зміщення сенсора смарт-

годинника й зменшення точності виявлення піків пульсу та інтервалів серцевих скорочень.

Таким чином, розроблений метод побудови вектору стану користувача у навчальній програмній системі забезпечує можливість моніторингу стану за допомогою даних з носимого пристрою (смарт-годинника) та формує вектор стану користувача у режимі soft real-time. Отриманий вектор може бути представлений у структурованому JSON – форматі для використання у міжмодульному інтерфейсі при вирішенні задачі диспетчеризації навчальних модулів навчальної програмної системи для персоніфікованої адаптації процесу навчання.

3.3 Розробка методу персоніфікованої адаптації процесу вибору навчальних модулів на основі векторів профілю та стану користувача

Сучасні підходи до вибору навчального контенту, зазвичай базуються на використанні статичних правил, або рекомендаційних алгоритмів колаборативної фільтрації, які працюють в режимі офлайн. Такі підходи не дають можливості оперативно підібрати навчальний модуль в залежності від комбінації базових навичок та поточного стану користувача. Проте дана задача має бути вирішена при швидких змінах стану користувача внаслідок стресу або втоми при навчанні, а також когнітивного перевантаження внаслідок недостатньої базової підготовки, що мають серйозний вплив на ефективність навчання. Дана проблема може виникнути навіть при початковому виборі рівня складності на основі профілю користувача, розглянутому в підрозділі 3.1. Тому розробка методів диспетчеризації модулів у адаптивних навчальних програмних системах, що враховують як психометричний профіль користувача, так і його поточний психофізіологічний стан, потребує свого вирішення.

Розроблений у даному розділі метод персоніфікованої адаптації процесу вибору навчальних модулів використовує представлені у підрозділах 3.1 та 3.2 вектори профілю користувача та стану користувача для підбору навчальних

модулів з бази сценаріїв, які були розглянуті у підрозділі 2.1. Диспетчеризація модулів виконується з урахуванням поточного рівня підготовки користувача, можливостей його адаптації відповідно до рівня когнітивного навантаження при навчанні, новизни сценарію навчання для користувача.

Існуючі підходи до адаптивної диспетчеризації навчальних модулів характеризуються наступними ключовими властивостями, які обмежують побудову адаптивних персоніфікованих програмних систем на основі комбінованої оцінки профілю та стану користувача.

По-перше, традиційний підхід до диспетчеризації модулів на основі статичних причинно-наслідкових правил щодо стану користувача навчальної програмної системи (наприклад, правил виду «якщо частота серцевих скорочень >120 ударів на хвилину, то знизити на 1 рівень складність навчального модуля») не враховує індивідуальні відмінності користувача. Зокрема, в залежності від рівня спортивної підготовки, базова частота серцевих скорочень може змінюватись від 50 до 80 ударів на хвилину, що потребує адаптації порогового значення в 120 ударів на хвилину у наведеному прикладі. Також в залежності від поточного рівня досвіду один користувач може набувати знань при роботі зі складними модулями навіть при підвищеному рівні серцебиття, тоді як інший навіть при типовій частоті серцевих скорочень потребує простих навчальних модулів внаслідок недостатньої підготовленості. Тому персоніфікована система, яка використовує лише статичні правила з пороговими значеннями частоти серцевих скорочень, може неточно змінити складність модуля. В результаті користувач програмної системи може пропустити важливі на поточному рівні підготовки навчальні теми.

По-друге, традиційні рекомендаційні алгоритми основі колаборативної фільтрації [167] або на основі правил [162] зазвичай використовуються для підбору статичного контенту у платформах дистанційного навчання, наприклад навчальних фільмів або онлайн-курсів. Проте такі підходи не призначені для вивчення знань та навичок, що пов'язані із безпекою людей і тому не враховують поточний стан користувача під час сесії навчання. Тобто вибір навчального

модуля виконується з урахуванням минулих уподобань користувачів з аналогічними профілями або на основі схожості семантики модулів, без урахування поточного рівня когнітивного навантаження або перенавантаження. Вибір навчальних модулів у такій ситуації може приводити до стресу та втоми внаслідок підвищеної складності модулів, що не відповідають стану користувача.

По-третє, методи адаптації складності навчальних модулів на основі порогових значень параметрів стану людини не враховують тривалість стану з підвищеною частотою пульсу. Наприклад, підвищення частоти серцевих скорочень на декілька секунд, що виникає внаслідок різкого руху користувача, та підвищена частота пульсу протягом хвилини-двох внаслідок когнітивного перевантаження при навчанні потребують різних схем вибору модулів. Також потрібно врахувати структурно-логічну схему курсу, новизну модуля, тобто перевіряти, чи виконував користувач схожі сценарії навчання раніше тощо. Наприклад навчання водінню за темою «Обгін на двосмуговій дорозі» потребує попереднього вивчення теми «Правила використання смуг руху». Без врахування послідовності вивчення модулів адаптивна система може помилково знизити складність модуля для поточного користувача. Тобто сукупність цих факторів приводить до зниження точності підбору навчальних модулів у порівняння з вибором людини-інструктора та відповідного зниження рівня задоволеності користувачів такої системи.

Таким чином, наведене протиріччя між практичною потребою з персоніфікованого підбору модулів в навчальній програмній системі та можливостями існуючих підходів щодо адаптивної диспетчеризації програмних модулів в реальному часі (або у м'якому реальному часі) підкреслює актуальність проблеми, що вирішується в даному підрозділі.

Проблематика диспетчеризації навчальних модулів з урахуванням психометричного профілю користувача та його поточного стану безпосередньо пов'язана з моделюванням динамічної поведінки користувача як вхідного параметра для прийняття рішень в режимі м'якого реального часу за умов

обмежень щодо обчислювальної складності та задоволення нефункціональних вимог до диспетчеризації. Останні пов'язані із затримкою часу та точністю рекомендацій щодо навчального модуля.

Диспетчер модулів, який виконує персоніфіковану адаптацію, представляє собою один із ключових компонентів розробленої у підрозділі 2.3 гібридної програмної архітектури. Він отримує вектори профілю та вектори стану користувача, виконує двоетапну фільтрацію модулів, сортує модулі за корисністю і передає навчальні модулі-кандидати для вибору користувачем. Логування (запис у журнал подій) процесу вибору модулів при персоніфікованій адаптації дає можливість в подальшому відтворити фактичну послідовність виконання даного процесу, що забезпечує можливість побудови циклу адаптації навчальної системи. Такий цикл базується на розробленому методі верифікації процесу підготовки водіїв в навчальних програмних системах автошкіл (підрозділ 3.4).

Аналіз літературних джерел щодо диспетчеризації модулів для адаптивних навчальних систем, показав, що існуючі підходи реалізують окремо підбір модулів наступними альтернативними способами:

- на основі історії взаємодії з користувачем;
- з урахуванням рівня компетентності користувача;
- з урахування поточного психофізичного стану користувача.

Дослідження [167] присвячено побудові рекомендацій на основі традиційної колаборативної фільтрації. Даний підхід забезпечує персоналізацію рекомендацій, однак використовує пакетну обробку і тому не призначений для роботи в режимі онлайн. Цей підхід не враховують біометричні дані користувача і не призначений для застосування в системах віртуальної реальності.

Робота [162] використовує статичні правила для адаптації складності навчальних модулів на основі результатів біометричного моніторингу. Але дане дослідження не враховує базові індивідуальні особливості користувачів, наприклад частоту серцевих скорочень у стані спокою, що може привести до помилок для користувачів з підвищеною базовою частотою пульсу.

Дослідження [148, 149] використовують психометричну модель IRT 3PL з логістичною функцією з трьох параметрів та САТ-підхід для того, щоб визначати рівень підготовки користувачів. Оцінка рівня підготовки виконується на основі ймовірності правильної відповіді з урахування рівня складності завдання. Однак ці підходи потребують виконання набору питань, який адаптується у процесі відповідей, що на практиці потребує 10-15 хвилин часу. Вказані підходи оцінюють рівень підготовки користувача, однак не враховують його поточний стан і тому не призначені для вибору модулів у процесі навчання.

Розроблений метод використовує вектор профілю користувача та вектор його поточного стану. Вектор профілю оновлюється після виконання порогової кількості завдань, наприклад, після 40 завдань або 2 тижнів навчання. Вектор поточного стану оновлюється кожен раз після проведення моніторингу стану користувача. Стан користувача оновлюється в режимі м'якого реального часу і містить біометричні дані користувача за актуальний період часу, зазвичай за останню хвилину. Представлена у векторах інформація передається у форматі JSON, що дає можливість стандартизувати інтерфейс між рівнями гібридної архітектури системи навчання.

Попередній відбір навчальних модулів виконується на основі послідовної фільтрації модулів за темою, складністю та тривалості навчання.

Фільтрація тем згідно попередніх вивчених навчальних сценаріїв дає можливість виключити із подальшого розгляду модулі, для яких користувач ще не вивчив обов'язкові попередні теми. Наприклад, перед вивченням модулю «Обгін на двосмуговій дорозі» попередньо користувач має вивчити теми «Дорожня розмітка» та «Правила обгону». Якщо користувач не вивчав ці теми, то модуль «Обгін на двосмуговій дорозі» виключається з розгляду.

Фільтр складності призначений для відхилення модулів, складність яких виходить за межі поточного рівня підготовки користувача. Тобто на даному кроці видаляються модулі, які явно можуть спричинити когнітивне перенавантаження користувача.

Фільтрація вивчених модулів призначена для виключення з розгляду тих модулів, які користувача вже вивчав декілька разів (тобто кількість повторів у вивченні досягла порогового значення). На даному кроці можуть бути винятки, до яких належать модулі безпечного водіння, наприклад «Екстрене гальмування».

Обчислення корисності модуля виконується згідно рівня компетентності користувача, а також з урахуванням когнітивного стану та новизни навчального сценарію. Новизна сценарію розраховується в залежності від кількості повторів у вивченні навчального модуля. Показник новизни зменшується від 1 до 0 в залежності від кількості повторних навчань користувача з використання поточного модуля.

Підмножина модулів-кандидатів з максимальною корисністю надаються користувачеві. З них він має відібрати один модуль. Результати вибору користувача логуються (записуються в журнал подій) для подальшого удосконалення процесу персоніфікованої адаптації навчання.

Метод персоніфікованої адаптації навчальних модулів визначає алгоритм роботи диспетчера модулів ModuleDispatch як інтеграційного компоненту адаптивної навчальної системи. Сукупність методів формування профілю користувача (підрозділ 3.1), оцінки поточного стану користувача (підрозділ 3.2), та адаптивної верифікації процесу вибору модулів на основі інтелектуального аналізу процесів (підрозділ 3.4) формує єдиний цикл підтримки персоналізованого навчання. У розробленій гібридній архітектурі «клієнт-периферія-хмара» (підрозділ 2.4) компонент диспетчера модулів розгортається на периферійному сервері у локальній мережі автошколи з тим, щоб зменшити затримки при взаємодії з користувачем, в тому числі за рахунок кешування даних. Такий підхід забезпечує автономність при відмові доступу до хмари або до глобальної мережі в цілому.

Розроблений метод має забезпечувати виконання наступних функціональних вимог:

- фільтрація модулів, які не відповідають рівню підготовки та стану користувача;

- обчислення ступеню корисності навчального модуля для користувача;
- ранжування і відбір навчальних модулів за корисністю.

Метод має задовольняти наступним нефункціональним вимогам:

- робота в режимі м'якого реального часу;
- забезпечення порогового рівня прийнятності користувачем.
- незалежність від процесу підбору модулів від особливостей предметної області.

Робота в режимі м'якого реального часу передбачає несуттєві затримки на диспетчеризацію, на практиці такі затримки мають бути не більше 1 секунди.

Рівень прийнятності запропонованого модуля для користувача розраховується в залежності від того, чи вибрав користувач перший модуль із списку рекомендованих.

Незалежність процесу персоніфікованої адаптації від особливостей предметної області реалізовано за рахунок використання векторів профілю та стану користувача. Розроблений метод може забезпечувати підтримку навчання у таких сферах, як водіння, медицина, авіаційна галузь, високотехнологічна промисловість.

Задача персоніфікованої адаптації навчальних модулів є багатокритеріальною задачею, яка враховує:

- відповідність рівня складності навчального модуля a_i рівню підготовки користувача на основі вектору профілю користувача V_{user}^1 ; вектор профілю користувача включає рівні складності модулів для користувача L_m згідно порогових значень здібностей користувача (3.7), поточну оцінку здібностей користувача b_j^* , профіль помилок U_{error} із частотами помилок (3.9) по рівням l , а також траєкторію U_{trend} зміни рівня підготовки користувача навчальної програмної системи, представлена лінійною регресією (3.11);

– відповідність рівня складності модулю a_i поточному стану користувача, представленому вектором V_{user}^2 ; вектор поточного стану містить HRV_{norm} – нормалізовану варіабельність серцевих скорочень (3.14), HR_{norm} – нормалізовану частота серцевих скорочень (3.16), поточний стан когнітивного навантаження, $B_{user} = \{Високе навантаження; Норма; Низьке навантаження\}$, а також мітку часу $t_{current}$;

– новизна $Nov(a_i)$ модулю a_i для користувача.

При виконанні задачі персоніфікованої адаптації використовуються три компоненти функції корисності: рівень складності модулю $L(a_i)$, рівень когнітивного навантаження $B_{user}(a_i)$, а також новизна модулю $Nov(a_i)$.

Корисність $Utility$ кожного з компонентів визначається з використанням ядра Гауса для першої компоненти $L(a_i)$, порогових значень для другої компоненти $B_{user}(a_i)$, а також через лінійну інтерполяцію для третьої компоненти $Nov(a_i)$.

Корисність $Utility(L_m, a_i)$ визначається на основі порівняння рівня складності модулю $L(a_i)$ та цільового значення складності L_m :

$$Utility(L_m, a_i) = \exp\left(-\left(\frac{L(a_i) - L_m}{2\sigma}\right)^2\right). \quad (3.19)$$

Складність навчального модулю $L(a_i)$ визначається у відповідності до (2.10) на основі атрибутів складності дорожнього сценарію, який представлений у модулі.

Використання ядра Гауса у (3.19) забезпечує згладжування для широкого діапазону рівнів. Тобто модулі із складністю близькою до цільової мають значення корисності близьким до 1.

Корисність модуля у відповідності до поточного стану користувача визначається на основі порогових значень складності модулю $L(a_i)$:

$$Utility(B_{user}, a_i) = \begin{cases} 1 \text{ iff } B_{user} = \text{Норма}, \\ 1 - L(a_i) \text{ iff } B_{user} = \text{Високе навантаження}, \\ L(a_i) \text{ iff } B_{user} = \text{Низьке навантаження}. \end{cases} \quad (3.20)$$

Якщо для поточного модулю a_i когнітивне навантаження є нормальним, то корисність є максимальною і дорівнює 1. У випадку високого навантаження корисність розраховується як $1 - L(a_i)$, тобто чим вище складність модулю, тим нижче корисність. У випадку низького навантаження корисність розраховується як $L(a_i)$, тобто чим вище рівень складності модулю, тим вище корисність.

Новизна розраховується як нормована різниця між максимально допустимою кількістю разів проходження модулів та кількістю виконаних спроб вивчення модулю $Used(a_i)$:

$$Nov(a_i) = \frac{MaxUsed - Used(a_i)}{MaxUsed}. \quad (3.21)$$

На практиці $MaxUsed$ зазвичай приймається рівним 3.

Загальна корисність модулю визначається як згортка компонент (3.19), (3.20) та (3.21):

$$Utility(a_i) = w_1 Utility(L_m, a_i) + w_2 Utility(B_{user}, a_i) + w_3 Nov(a_i). \quad (3.22)$$

Ваги w_1 , w_2 та w_3 визначаються емпірично на основі історії навчання користувачів.

Модулі a_i^* при диспетчеризації підбираються за максимальним значенням функції корисності:

$$a_i^* = \arg \max_i (Utility(a_i)). \quad (3.23)$$

Метод персоніфікованої адаптації процесу навчання охоплює наступні етапи [4].

Етап 1. Попередня фільтрація модулів.

На першому етапі послідовно використовуються три фільтри відбору підмножини модулів. Фільтри базуються на правилах, що задають обмеження щодо відповідності тематики модуля, складності модулів та кількості повторів при навчанні.

Крок 1.1. Відбір модулів за темою.

На першому кроці етапу відбираються модулі, які належать до заданої підмножини тем згідно структурно-логічної схеми курсу. Результатом кроку є $A_{filtered}^1$.

Крок 1.2. Фільтрація модулів за рівнем складності.

На другому кроці відбираються підмножина модулів $A_{filtered}^2$, що знаходяться у заданому діапазоні складності ΔL :

$$A_{filtered}^2 = \{a_i \in A_{filtered}^1 : |L(a_i) - L_m| \leq \Delta L\}. \quad (3.24)$$

Крок 1.3. Відбір модулів за кількістю повторних проходжень.

На третьому кроці відсіюються модулі, які користувач вже опановував задану (порогову) кількість разів $MaxUsed$.

$$A_{filtered}^3 = \{a_i \in A_{filtered}^2 : Used(a_i) < MaxUsed\}. \quad (3.25)$$

Етап 2. Обчислення значення елементів функції корисності для відібраних модулів.

На другому етапі виконується комплексна оцінка корисності модулів із множини $A_{filtered}^3$ для користувача, яка враховує відповідність рівню підготовки користувача, поточному стану користувача, а також рівень новизни модуля для користувача.

Крок 3.1. Перша складова обчислюється згідно (3.19) та враховує профіль користувача із вхідних даних.

Крок 3.2. Друга складова обчислюється згідно (3.20) та враховує поточний стан курсанта автошколи.

Крок 3.3. Третя складова дає можливість врахувати кількість повторів при вивченні модулів. У відповідності до (3.21) нові модулі отримують перевагу перед тими, які користувач вже вивчав.

Крок 3.4. Загальна корисність модулю визначається згідно (3.22).

Етап 3. Відбір модулів із максимальним значення корисності для представлення користувачеві.

На третьому етапі виконується ранжування модулів за значенням корисності $Utility(a_i)$, обчисленим за (3.22). Із отриманої упорядкованої підмножини відбираються модулі із максимальним значення корисності для користувача. Наприклад, вибираються 3 найкращих модуля.

Етап 5. Вибір модуля користувачем.

На даному етапі користувачеві представляються відібрані модулі з найбільшим значенням корисності. Користувач має вибрати один із модулів. Якщо за заданий інтервал часу, наприклад 30 секунд, користувач не вибрав модуль, то відібраним вважається навчальний модуль з максимальним значенням корисності.

Результати вибору користувача логуються для подальшого аналізу.

Розроблений метод персоніфікованої адаптації навчальних модулів на основі оцінки профілю та стану користувача має наступні переваги.

По-перше, комплексна оцінка забезпечує підбір складності модуля згідно рівня підготовки та поточного стану користувача. На відміну від розробленого підходу, традиційно адаптивні системи навчання використовують профіль

користувача, який відображає його рівень підготовки на основі історії виконаних завдань. Проте даний профіль не містить інформації щодо поточного психофізіологічного стану користувача. Альтернативний підхід до оцінки рівня користувача на основі поточних біометричних даних (наприклад, частоти серцебиття) не враховує базовий рівень його знань. В результаті користувач з низьким рівнем стресу може отримати для вивчення занадто складний модуль, який не зможе опанувати. Або ж підготовлений користувач у стресовому стані отримає занадто простий модуль.

По-друге, розроблений метод послідовно виконує фільтрацію модулів на основі правил, після чого виконує підбір модулів лише для відфільтрованої підмножини. Такий підхід характеризується суттєвим зменшенням обчислювальних витрат порівняно із традиційним підходами на основі колаборативної фільтрації та матричної факторизації.

В-третьє, розроблений метод підтримує взаємодію з користувачем шляхом вибору одного із запропонованих модулів-кандидатів, що сприяє підвищенню задоволеності користувача. Зокрема, користувач може врахувати поточні обмеження свого часу при виборі модуля.

В-четвертих, використання правил попередньої фільтрації на першому етапі методу створює умови для побудови в подальшому пояснень щодо персональних причин підбору конкретного модулю для користувача. Такий підхід відрізняється від традиційних підходів на основі машинного навчання, в яких вибір є непрозорим для користувача.

Особливо важливим пояснення є у випадку, якщо потрібна сертифікація навчальної системи. Така сертифікація зазвичай проводиться як у транспортній галузі.

Також слід зазначити, що існуючі методи диспетчеризації як правило реалізуються у вигляді монолітних модулів без відповідного API, що ускладнює їх інтеграцію з мікросервісами при побудові гібридної програмної архітектури, оскільки в основі таких мікросервісних архітектурних підходів лежать принципи слабого зв'язку між модулями та високої зв'язності модулів. Розроблений метод

використовує вектори вхідних даних, представлені у JSON –форматі, що дає можливість інтегрувати метод з мікросервісним рівнем в розроблений гібридний програмний архітектурі з периферійними обчисленнями.

Обмеження розробленого методу визначають умови його застосування та можливості подальшого удосконалення.

По-перше, метод залежить від коректності метаданих у модулях, що містять сценарії навчання водінню (кількість об'єктів сценарію, передумови використання).

По-друге, при підборі модулів метод враховує поточний стан користувача, але не враховує такі довгострокові темпоральні патерни, як циркадні ритми, криву забування тощо. Останні можуть суттєво впливати на когнітивні можливості курсанта автошколи.

В-третьє, метод використовує набір даних про виконання модулів для уточнення метаданих кожного модуля. Однак такий підхід не враховує проблему холодного старту, тобто проблему налагодження нових модулів. Подальший розвиток методу з урахуванням проблеми холодного старту може бути пов'язаний із використанням метаданих із схожих модулів, наприклад схожих сценаріїв, погодних умов, рівня трафіку. Такі дані можуть бути отримані також із датасетів, що фіксують процес водіння.

Таким чином, розроблений метод персоніфікованої адаптації процесу навчання забезпечує підбір модуля згідно рівня підготовки та поточного стану користувача із можливістю вибору одного із запропонованих модулів користувачем.

3.4 Розробка методу верифікації процесу вибору навчальних модулів з використанням технології інтелектуального аналізу процесів

Верифікація адаптивної логіки програмних систем, що змінюють свою поведінку в залежності від рівня знань та стану користувача, є ключовою задачею в навчальних системах з підвищеними вимогами щодо безпеки, зокрема

в фінансових, медичних, транспортних, в тому числі авіаційних навчальних системах. Проте існуючі підходи до верифікації програмних систем мають ряд суттєвих обмежень при перевірці адаптивної логіки в умовах експлуатації системи навчання з різними профілями та станами користувачів.

По-перше, традиційна статична формальна верифікація з використанням методів *model checking* призначена для перевірки правильності алгоритмів в першу чергу на етапі розробки програмної системи. Дана верифікація, зокрема, реалізована в інструментальному засобі NuSMV [168] і полягає в перевірці властивостей всіх станів, в які система може перейти. Але в адаптивних системах потрібно перевірити зміни фактичної поведінки в залежності від реальних значень параметрів профілю та стану користувача. Тому статична верифікація обмежує виявлення помилок при адаптації поведінки програмної системи, зокрема у випадках граничних комбінацій вхідних параметрів, які можуть виникати при роботі програмної системи. Також статична верифікація не враховує порогові значення рівня підготовки й стану користувачів при персоніфікованому підключенні навчальних модулів, що може приводити до затримок при адаптації згідно векторів профілю та стану користувача.

По-друге, підходи на основі моніторингу інваріантів виконують локальну перевірку властивостей системи з використанням темпоральних операторів, але такі підходи не забезпечують повного аналізу можливих траєкторій процесу персоніфікованої адаптації програмної системи, що не дає можливість виявити порушення останніх.

По-третє, *offline*-тестування з використанням відомих сценаріїв зазвичай виконується на синтетичних наборах даних і тому аналізує лише обмежений набір сценаріїв адаптації, що не дає можливість відобразити реальний діапазон станів користувачів і, відповідно, виявити граничні випадки поведінки системи.

Таким чином, існуючі підходи не використовують зворотний зв'язок на основі перевірки відповідності правил диспетчеризації модулів і траєкторій поведінки, які зафіксовані в логах системи, та стану й рівня підготовки користувачів. Даний недолік може привести до підключення складних модулів

при поточному когнітивному перевантаженні користувача, відсутності реакції системи навчання на зміну поточного стану користувача та неможливості автоматизованої адаптації параметрів навчання на основі відхилення використаних правил від траєкторії навчання для персоніфікації навчального процесу.

Персоніфікована адаптація має забезпечувати генерацію множини варіантів конфігурації навчального середовища, тобто множини можливих навчальних модулів для користувача. Контроль коректності цих адаптованих траєкторій навчання курсанта автошколи дає можливість удосконалити процес навчання, оскільки неточне налаштування правил персоніфікованої адаптації навчальних модулів знижує ефективність підготовки користувача програмної системи.

У даному підрозділі розроблено метод верифікації адаптивної програмної системи навчання, який включає дві ключових фази. По-перше, з використанням технології process mining (PM) [169] формується модель фактичного процесу персоніфікованої адаптації програмної системи безпосередньо з логу (журналу подій), що містить всі виконані траєкторії навчання. По-друге, виконується перевірка відповідності отриманої з логу фактичної моделі та заданих правил адаптації. На основі оцінки невідповідності правила персоніфікованої адаптації можуть бути удосконалені [5, 9].

Журнал подій представляє собою послідовність записів про всі події, що відбуваються при виконанні персоніфікованої адаптації процесу навчання. Цей журнал фіксує зв'язок між поточним вектором стану курсанта і рішенням модуля-диспетчера у кожен момент часу у вигляді послідовності подій e_i , кожна з яких має вигляд кортежу з різнотипних елементів:

$$e_i = \langle t_i, e_type_i, u_id_i, L_m^i, \theta^i, B_s^i, Cg^i, Md_j^i, act_i, sts_i \rangle, \quad (3.26)$$

де t_i – мітка часу;

e_type_i – тип події, що відображає підключення, відключення модулю, статус користувача тощо;

u_id_i – ідентифікатор користувача;

L_m^i – поточний рівень підготовки користувача;

θ^i – відповідність рівня підготовки користувача рівню поточного навчального модулю;

B_s^i – стан користувача;

Cg^i – рівень когнітивного навантаження користувача (високий, нормальний, низький);

Md_j^i – навчальний модуль;

act_i – дія з диспетчеризації модуля (підключення, відключення, оновлення);

sts_i – результат дії.

Журнал формується і доповнюється у кожній сесії навчання і використовується при вирішенні першої задачі інтелектуального аналізу процесів (process mining) – задачі виявлення процесів.

В даному підрозділі для вирішення цієї задачі використовується алгоритм Inductive Miner [170]. Він виконує ієрархічну декомпозицію журналу на підпоследовності. Потім алгоритм рекурсивно виявляє структурні шаблони процесу адаптації: последовності, вибір альтернатив, цикли, паралельні гілки процесу. Результат аналізу представляє собою ієрархічну модель процесу. Дана модель процесу формально представляється як мережа Петрі G :

$$G = (St, T, E), E \subseteq St \times T \times St, \quad (3.27)$$

де $St = \{st^i\}$ – множина станів процесу;

T – множина переходів, що відображають персоніфіковану адаптацію модулів;

E – множина дуг мережі Петрі.

Кожен стан st_i задається через показники стану користувача:

$$st^i = \langle L_m^i, B_s^i, Cg^i \rangle_i. \quad (3.28)$$

Переходи відповідають діям диспетчера з підключення або відключення модулів. Кожна дуга визначає перехід між станами за допомогою дії з персоніфікованої адаптації.

Вибір алгоритму Inductive Miner для побудови моделі фактичного процесу адаптації його перевагами над альтернативним алгоритмами. Зокрема, популярний алгоритм Alpha Miner [171] не обробляє цикли довжиною 1 і 2 та може некоректно відтворювати паралельні гілки при неповних журналах. Такі обмеження приводять до побудови помилкових моделей в адаптивній системі із змінними траєкторіями навчання. Ефективний алгоритм Heuristics Miner [172] враховує частоти переходів між станами у процесі адаптації, однак побудовані ним моделі можуть містити тупикові ситуації, що унеможлиблює подальшу перевірку відповідності між моделями фактичного та цільового процесів адаптації.

Цільова модель процесу адаптації створюється із правил диспетчеризації і визнає точну послідовність подій з адаптації у часі з урахуванням поточних значень профілю та стану курсанта автошколи.

Правило вибору навчального модуля має вигляд:

$$\begin{aligned} \forall \tau: & \left(L_m^i(t) \geq C(M_j) \wedge B_s^i(t) \leq B^*(M_j) \wedge Cg^i(t) \neq \text{HIGH} \right) \\ \Rightarrow & Use(M_j, t), \end{aligned} \quad (3.29)$$

де $L_m^i(\tau)$ – поточний рівень навичок у момент τ ;

$C(M_j)$ – необхідний рівень навичок для підключення модуля M_j ;

$B_s^i(\tau)$ – біометричний показник стресу курсанта у момент τ ;

$B^*(M_j)$ – допустиме значення стресу для підключення модуля M_j ;

$Cg^i(\tau) \neq \text{HIGH}$ – умова відсутності критичного когнітивного навантаження для користувача навчальної системи;

$Use(M_j, t)$ – вибір на використання навчального модуля M_j .

Якість еталонної моделі процесу адаптації безпосередньо впливає на результат верифікації, оскільки неповнота правил приводить до хибних спрацювань, а протиріччя в правилах не дають можливості коректне відтворити траєкторії адаптації. Еталонна модель також може бути сформована у вигляді мережі Петрі.

Перевірка відповідності еталонної та фактичної моделей виконується з використанням показників відповідності та точності. Показник відповідності $Fitness(P_{fact}, P_{ref})$ характеризує частку траєкторій логу, які повністю відтворюються в цільовій моделі процесу адаптації:

$$Fitness(P_{fact}, P_{ref}) = \frac{Traces_{correct}}{Traces}, \quad (3.30)$$

де P_{fact}, P_{ref} – фактична та цільова моделі процесу адаптації;

$Traces_{correct}$ – кількість коректно відтворених у цільовій моделі трас із фактичної моделі;

$Traces$ – кількість трас у фактичній моделі.

Значення $Fitness(P_{fact}, P_{ref}) = 1$ свідчить про повну відповідність фактичної та цільової моделей.

Показник точності $Precision(P_{fact}, P_{ref})$ визначає частку переходів із цільової моделі, які відображені в журналі подій:

$$Precision(P_{fact}, P_{ref}) = \frac{|T_{fact}|}{|T|}, \quad (3.31)$$

де T_{fact} – множина відображених у журналі подій переходів;

T – множина всіх переходів в моделі.

Значення $Precision(P_{fact}, P_{ref}) = 1$ свідчить про те, що модель P_{ref} не містить надлишкових переходів (дій з підключення або відключення модулів при адаптації), які не були реалізовані у фактичній моделі.

Виявлення відхилень у реалізації адаптивної логіки здійснюється після перевірки відповідності цільової та фактичної моделей з тим, щоб ідентифікувати події логіки, які не відображені у еталонній моделі:

$$Violat(e_i) = True \text{ if } \exists e_i : e_i \in P_{fact} \wedge e_i \notin P_{ref}. \quad (3.32)$$

де e_i – подія логіки.

Порушення $Violat(e_i)$ означає, що не виконується умова (3.29), тобто навчальний модуль був призначений без урахування поточного рівня знань або поточного стану користувача навчальної програмної системи.

Порушення класифікуються за рівнями критичності згідно їх потенційного впливу на якість підготовки користувача системи навчання:

– CRITICAL, коли відбувається підключення складного модуля при критичному когнітивному навантаженні та підвищеному біометричному показнику стресу, тобто повністю не виконується умова (3.29);

– HIGH, коли відбувається підключення навчального модуля зі складністю, яка суттєво перевищує рівень знань та навичок курсанта, тобто не виконується умова $L_m^i(t) \geq C(M_j)$;

– MEDIUM, коли система навчання із затримкою враховує зміну вектору стану користувача, тобто коли не виконується умова $B_s^i(t) \leq B^*(M_j)$;

– LOW, що відображає несуттєві порушення часових обмежень.

Розроблений метод включає етапи формування журналу подій, побудови моделей процесу персоніфікованої адаптації, перевірки відповідності фактичної

та цільової моделей процесу адаптації та виявлення відхилень у реалізації адаптивної логіки.

Етап 1. Логування подій адаптації програмної системи.

Крок 1.1. Впровадження механізму логування в системі навчання. Механізм логування впроваджується із застосуванням аспектно-орієнтованого програмування, тобто шляхом додавання функціональності протоколювання без зміни логіки основного коду через «поради» [173]. Така додаткова логіка виноситься в окремий модуль. Порада може виконуватись до та після виконання основного коду. Механізм реалізації порад залежить від мови програмування. Зокрема, може бути використаний патерн Декоратор.

Крок 1.2. Формування записів логу.

Запис для подій зміни стану курсанта автошколи, а також підключення або відключення модулів формується у форматі, визначеному виразом (3.26).

Крок 1.3. Формування логу.

Результуючий лог формується у стандарті XES, який задає xml – формат представлення подій. Дані у стандарті XES без додаткових перетворень обробляються алгоритмами process mining, зокрема на платформі ProM.

Етап 2. Побудова цільової моделі процесу адаптації.

На даному етапі формується множина правил диспетчеризації навчальних модулів з урахуванням векторів профілю та стану користувача програмної системи. Кожне з правил визначає перехід між станами програмної системи навчання. Правила вибору навчальних модулів представляються у вигляді формул лінійної темпоральної логіки аналогічно (3.29) – для кожного навчального модуля визначаються умови підключення й відключення в залежності від поточних значень вектору стану.

Сукупність правил формує модель цільового процесу адаптації у формі мережі Петрі. Для перетворення правил у мережу Петрі послідовно використовується бібліотека SPOT та інструмент синтезу мережі Petriify.

Етап 3. Виявлення фактичного процесу диспетчеризації модулів з використанням індуктивного алгоритму process mining.

Крок 3.1. Попередня обробка логу.

На першому кроці виконується фільтрація незавершених траєкторій, видалення можливих дублікатів та нормалізація міток часу.

Крок 3.2. Формування фактичної моделі процесу персоніфікованої адаптації. На даному кроці використовується алгоритм Inductive Miner із реалізацією у бібліотеці PM4Py [174]. Даний алгоритм виявляє шаблони складових процесу (послідовності подій, альтернативи, цикли та паралельні гілки виконання). Потім із цих шаблонів формується модель процесу (3.27).

Етап 4. Перевірка відповідності цільового та фактичного процесів адаптації програмної системи.

На даному етапі кожна траєкторія із журналу подій відтворюється на цільовій моделі і перевіряється наявність відповідного переходу в мережі Петрі.

Обчислюються показник відповідності (3.30) та показник точності (3.31) шляхом порівняння цільової та фактичної моделей процесу адаптації системи навчання.

Етап 5. Виявлення відхилень у реалізації адаптивної логіки.

Крок 5.1. Виявлення відхилень на основі правила (3.32) для таких обмежень:

- за рівнем навичок: $L_m^i(t) \geq C(M_j)$;
- за біометричними показниками: $B_s^i(t) \leq B^*(M_j)$;
- за рівнем когнітивного навантаження: $Cg^i(t) \neq \text{HIGH}$.

Крок 5.2. Виявлені порушення розподіляються за чотирма рівнями критичності (CRITICAL, HIGH, MEDIUM, LOW). Умови цих порушень задаються через показники стану користувача (3.28).

Практична перевага розробленого методу верифікації полягає в тому, що програмна система адаптивного навчання в автошколі має постійно приймати рішення щодо того, який навчальний сценарій (навчальний модуль) підключити у поточний момент конкретному користувачу. Рішення щодо підключення модулів мають враховувати рівень підготовки курсанта та його поточний стан.

Рівень підготовки для автошкіл свідчить про те, що курсант вже вміє як водій автомобіля, наскільки впевнено він виконує поточні вправи і чи готовий він до складнішого завдання. Поточний стан курсанта показує: чи він перевтомився, або є занадто збудженим і нездатним в поточний момент якісно засвоїти новий матеріал. Перша умова забезпечує адаптацію рішення щодо вибраних навчальних модулів під профіль знань користувача, а другу – персоніфікацію рішення згідно його психоемоційного стану, що у сукупності забезпечує персоніфіковану адаптацію навчання. Проте правила підключення модулів можуть бути неповними, що може привести до підключення занадто складного навчального модулю непідготовленому курсанту або ж запуску складного завдання в той момент, коли користувач у стані перенавантаження і не здатний опрацювати це завдання. Такі помилки збільшують витрати часу та знижують ефективність навчання, що приводить до невпевненості у складних ситуаціях при водінні.

Метод верифікації вирішує задачу перевірки коректності адаптивних рішень програмної системи навчання і виявляє правила, що порушують персоніфіковану адаптацію до рівня знань та поточного стану користувача.

Правила верифікації мають такий вигляд. Програмна система адаптивного навчання не повинна підключати складний модуль у випадку, якщо рівень навичок курсанта нижчий за необхідний мінімум. Таке підключення приводить до порушення адаптивності. Також система не повинна підключати будь-який модуль, якщо рівень стресу курсанта перевищує допустиму межу, оскільки це порушує персоніфікацію навчання. Програмна система не має підключати складний сценарій навчання у випадку, коли одночасно зафіксовано як стресовий стан, так і недостатній рівень підготовки, оскільки це порушення персоніфікованої адаптації в цілому.

Виявлені порушення розподіляються за критичним, серйозним, помірним та незначним рівнями. Критичні порушення виникають, коли є загроза здоров'ю курсанта під час заняття. Серйозні характеризуються порушення педагогічної логіки, коли навчання стає малоефективним. Помірні порушення виникають

коли система навчання реагує із запізненням на погіршення стану курсанта. Незначні порушення представляють дрібні технічні затримки, що погіршують роботу системи навчання.

Перевага розробленого методу базується на використанні технології process mining для порівняння траєкторій підключення модулів. Даний підхід дає можливість проводити ітеративну перевірку при доповненні фактичної моделі процесу персоніфікованої адаптації по мірі збільшення кількості слідів процесу у журналі подій.

Обмеження розробленого методу пов'язані із залежністю від якості логу. Пропуск при записі подій в журнал може привести до пропущених порушень у правилах персоніфікованої адаптації.

Таким чином, метод верифікації забезпечує умови для подальшого удосконалення процесу персоніфікованої адаптації підготовки у програмних навчальних системах.

У перспективі, вдосконалення програмної частини навчальних систем може бути реалізоване за рахунок використання хмарних сервісів, зокрема, у форматі приватної хмари для кожної особи, яка навчається, тобто створення так званої Private Cloud Groupe.

Висновки до розділу 3

1. Удосконалено метод побудови вектору профілю користувача навчальної програмної системи. Метод відрізняється від існуючих тим, що виконує оцінку латентних здібностей користувача з використанням психометричних моделей, визначає на цій основі рівні складності задач для користувача, опис типових помилок і формує вектор профілю користувача як формальну специфікацію інтерфейсу з користувачем. Метод дає можливість автоматизувати процеси персоніфікованої адаптації та верифікації модулів навчальної програмної системи з урахуванням значень компонентів вектору профілю користувача.

3. Удосконалено метод побудови вектору поточного стану користувача навчальної програмної системи. Метод на відміну від існуючих виконує аналіз варіабельності серцевого ритму, визначає патерни серцевого ритму та формує вектор поточного стану користувача, який може бути представлений в рамках програмного інтерфейсу для автоматизованої диспетчеризації модулів навчальної програмної системи. Метод забезпечує можливість інтеграції компоненти оцінки стану у програмну архітектуру адаптивної навчальної системи та подальшого персоніфікованого підбору навчальних модулів.

4. Вперше запропоновано метод персоніфікованої адаптації процесу навчання шляхом вибору навчальних модулів на основі векторів профілю та стану користувача. Метод, на відміну від існуючих, враховує рівень підготовки, поточний психофізичний стан користувача, а також ступінь новизни завдань у навчальних модулях і використовує попередню фільтрацію модулів за ступенем відповідності рівню підготовки користувача. Метод дає можливість скоригувати процес навчання з урахуванням як рівня знань, так і когнітивного навантаження користувача навчальної програмної системи.

5. Набув подальшого розвитку метод верифікації адаптивної програмної системи з використанням технології process mining. Метод включає етапи побудови логу адаптивної навчальної програмної системи, побудову моделей цільового та фактичного процесів персоніфікованої адаптації, виявлення порушень адаптивної логіки на базі порівняння цільової та фактичної моделей процесу персоніфікованої адаптації. Метод створює умови для удосконалення процесу персоніфікованої адаптації навчальної програмної системи з урахуванням різниці між моделями цільового та фактичного процесу навчання.

4 ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА МОДЕЛІ ТА МЕТОДІВ ПЕРСОНІФІКОВАНОЇ АДАПТАЦІЇ ПРОЦЕСУ ПІДГОТОВКИ ВОДІЇВ В НАВЧАЛЬНИХ ПРОГРАМНИХ СИСТЕМАХ АВТОШКІЛ

4.1 Реалізація та експериментальна перевірка гібридної архітектури навчальної програмної системи автошколи

Експериментальна перевірка методів персоніфікованої адаптації процесу навчання в автошколі, розроблених у розділах 3.1–3.4, потребує реалізації прототипу гібридної програмної архітектури, наведеної у підрозділі 2.3. Дана архітектура поєднує три різних патерни взаємодії модулів навчальної програмної системи:

- обробку в межах єдиного процесу на периферійному рівні, що реалізована для модулів, які реалізують критичний шлях адаптації процесу навчання;
- синхронна мережева взаємодія між клієнтом і периферією в локальній мережі автошколи, яка виконується для передачі векторів профілю та поточного стану користувача;
- асинхронний обмін інформацією про події між периферійним та мікросервісним рівнями архітектури, який призначений для виконання аналітичних функцій за межами критичного шляху.

Поєднання цих патернів дає можливість задовольнити як вимогу до зменшення затримки циклу персоніфікованої адаптації, так і вимогу до обмеження сукупної вартості програмної системи навчання.

Клієнтський рівень моделюється генератором навантаження Locust. Це відкрите програмне забезпечення на Python, призначене для навантажувального тестування. Код на Python описує сценарій поведінки одного віртуального користувача. За заданим сценарієм Locust відправляє HTTP-запити від багатьох віртуальних користувачів. Тобто Locust імітує одночасні сесії від декількох користувачів. Периферійний рівень містить сервер, що послідовно реалізує

методи побудови векторів профілю й стану користувача а також методи персоніфікованої адаптації процесу навчання. Мікросервісний рівень містить модуль верифікації адаптивної програмної навчальної системи. Взаємодія з периферійним рівнем організована за допомогою асинхронної черги повідомлень [10]. Реалізація розробленої гібридної програмної архітектури представлена в табл. 4.1.

Реалізація гібридної архітектури виконана за принципом контейнеризації, коли модулі функціонують як ізольовані програмні компоненти зі своїм середовищем виконання, а також бібліотеками. Кожен контейнер має свою конфігурацію, яка не залежить від поточних налаштувань вузла обробки даних. Координація реалізована за допомогою Docker Compose, відкритою системою оркестрації. Бажаний стан інфраструктури задається в єдиному конфігураційному файлі.

При проведенні експериментальної перевірки обмеження на затримку циклу персоніфікованої адаптації встановлено в 250 мс. Цей цикл рахується від моменту отримання запиту від користувача периферійним сервером й до моменту надання користувачеві персоніфікованого навчального модуля. Обґрунтування заданого порогу 250 мс є таким. Персоніфікована адаптація при перевищенні порогу сприймається користувачем як «загальмованість» системи, що приводить до зниження ефективності навчання.

Реалізація гібридної архітектури охоплює використання продуктів з відкритим вихідним кодом з ліцензіями вільного використання, модифікації та розповсюдження: FastAPI; Apache Kafka; Jaeger; Prometheus; Grafana; PyTorch.

FastAPI представляє собою фреймворк для побудови HTTP-інтерфейсів мовою Python. Фреймворк надається за ліцензією MIT. Фреймворк містить шаблонний код з маршрутизації запитів, валідації даних, а також серіалізації відповідей. Ключова перевага FastAPI для використання у навчальній системі полягає у застосуванні асинхронної моделі. Фреймворк обробляє вхідні запити паралельно, не блокуючи обробку під час очікування відповіді від кожного з модулів, що дозволяє зменшити затримку обробки.

Таблиця 4.1 – Реалізація розробленої гібридної програмної архітектури

Модуль	Рівень гібридної програмної архітектури	Використані мови програмування, бібліотеки та фреймворки	Призначення/реалізований метод
edge-server	Периферійний	FastAPI	Підтримка оркестрації методів, розроблених у підрозділах 3.1–3.3
cat-service	Периферійний	Python, pyirt	Метод формування профілю користувача (підрозділ 3.1)
lstm-service	Периферійний	Python, PyTorch	Метод оцінки поточного стану користувача (підрозділ 3.2)
map-service	Периферійний	Python	Метод персоніфікованої адаптації (підрозділ 3.3)
kafka	Мікросервісний (асинхронна взаємодія)	Apache Kafka	Асинхронна черга подій між рівнями
process-mining	Мікросервісний	Python, PM4Py	Метод верифікації адаптивної програмної навчальної системи (підрозділ 3.4)
postgres	Інфраструктура	postgres:15-alpine	Журнали навчальних подій, профілі здобувачів
jaeger	Інфраструктура	Jaeger All-in-One (версія 1.52)	Розподілене трасування циклу
prometheus	Інфраструктура	Prometheus((версія 2.48.0)	Збір метрик продуктивності
grafana	Інфраструктура	Grafana (версія 10.2.0)	Візуалізація результатів моніторингу персоніфікованої адаптації процесу навчання
Locust	Клієнтський	Locust	Генератор навантаження, що імітує запити від користувачів програмної системи

Apache Kafka – це платформа розподіленого обміну повідомленнями, ліцензована за Apache License 2.0. Kafka використовує патерн «видавець–передплатник». За даним патерном відповідна компонента системи може опублікувати повідомлення в топіку (іменованому каналі), а інші компоненти незалежно можуть зчитати ці повідомлення. Платформа Kafka реалізує асинхронний обмін між периферійним і мікросервісним рівнями. Обмін відбувається таким чином: сервер публікує подію, що послідовність персоніфікованої адаптації завершено і не очікує подальшої обробки цієї події мікросервісами аналітики. Завдяки цьому затримка на аналітику виключається з критичного шляху адаптації процесу навчання і виконується ізоляція мікросервісного рівня у гібридній архітектурі.

Система розподіленого трасування Jaeger фіксує траєкторію кожного запиту через компоненти програмної системи. Траєкторія має вигляд дерева трасування, в якому кожна операція описана як структурований запис із атрибутами: назва операції, час початку і час завершення, ідентифікатор здобувача, стан когнітивного навантаження тощо. Послідовність цих інтервалів показує, скільки часу зайняло виконання кожного з модулів на поточному шляху персоніфікованої адаптації процесу навчання.

Система моніторингу Prometheus на відміну від Jaeger через рівні проміжки часу записує інтегральні показники роботи програмної системи: кількість запитів від користувачів за секунду, розподіл затримок на критичному шляху, кількість активних сесій користувачів тощо. Ці дві системи вирішують задачі аналізу, як діагностики конкретного циклу персоніфікованої адаптації (Jaeger), так і оцінки загальної поведінки програмної системи (Prometheus).

Інструмент візуалізації даних Grafana може підключатись до Prometheus і Jaeger і відображати їх дані у вигляді панелей моніторингу. Система Grafana забезпечує спостереження за ходом експериментальної перевірки.

Фреймворк глибокого навчання PyTorch призначений для побудови й навчання штучних нейронних мереж.

Python-бібліотека IRT (Item Response Theory) призначена для моделювання взаємодії між учнем на системою навчання і призначена для оцінки здібностей та рівня підготовки користувача й складності навчальних завдань згідно IRT-моделей.

Відкрита бібліотека процес-майнінгу для Python PM4Py (Process Mining for Python) призначена для обробки журналів подій (логів), побудови процесних моделей із цих логів з використанням таких алгоритмів, як α -алгоритм, Inductive Miner, Heuristic Miner тощо, та для перевірки відмінностей між цільовою та фактичною процесною моделями.

Docker-образ на базі Alpine Linux 15-alpine реалізує контейнер з невеликим розміром образу.

PostgreSQL представляє собою об'єктно-реляційну систему управління базами даних з відкритим кодом. В гібридній архітектурі ця СУБД розміщена в контейнері postgres:15-alpine.

Взаємодія між рівнями гібридної архітектури реалізована через дві мережі, що логічно відокремлені, `edge_net` і `micro_net`. Периферійний сервер підключений до обох мереж і слугує шлюзом.

Ключовий фрагмент конфігурації з двох мереж наведено на рис. 4.1.

```
networks:
  edge_net:    # клієнт/периферія (синхронний виклик)
    driver: bridge
    ipam:
      config:
        - subnet: "172.20.0.0/16"
  micro_net:   # периферія / мікросервіси (асинхронна взаємодія)
    driver: bridge
    ipam:
      config:
        - subnet: "172.21.0.0/16"
```

Рисунок 4.1 – Мережева конфігурація

На рис. 4.1 представлено дві підмережі з адресними просторами 172.20.* та 172.21.*, трафік між якими не маршрутизується безпосередньо. Зовнішній

запит може досягти модулів мікросервісного рівня лише через периферійний сервер.

Вимірювання характеристик програмної системи здійснюється на трьох рівнях. Виконується розподілене трасування системою Jaeger, збір показників системою Prometheus, а також їх візуалізація за допомогою Grafana. Вимірювання затримки виконується перехоплювачем запитів (рис. 4.2)

```
t_start = time.perf_counter_ns()
response = await call_next(request)          # виконання повного циклу
t_edge_ms = (time.perf_counter_ns() - t_start) / 1_000_000

span.set_attribute("adaptation.nfr_r1_compliant", t_edge_ms <= 200.0)
response.headers["X-Edge-Latency-Ms"] = f"{t_edge_ms:.2f}"
```

Рисунок 4.2 – Вимірювання затримки

При вимірювання затримки спочатку фіксується час обробки запиту `t_start`. Після виконання `await` відповідь вже повністю сформована, оскільки закінчився процес адаптації. Змінна `t_edge_ms` містить затримку периферійного сервера у мілісекундах. `span.set_attribute` має істинне значення, якщо час виконання не перевищує 200 мс. Значення серверної складової затримки `t_edge_ms` включається у заголовок HTTP «X-Edge-Latency-Ms». Дана інформація дозволяє генератору Locust обчислити мережну затримку як різницю між повним часом відповіді і наведеним у заголовку значенням.

Збір біометричного сигналу зі смарт-годинника виконується постійно на клієнтському рівні у фоновому режимі поки користувач навчається. Попередня обробка біометричних даних передувє побудові вектору поточного стану користувача. Сенсорним пристроєм виступає смарт-годинник типу Wear OS або Samsung Galaxy Watch, що містить вбудований фотоплетизмографічний сенсор. Даний сенсор вимірює зміну об'єму крові в капілярах з кожним ударом серця.

Дані з годинника передаються через інтерфейс BLE 5.0. Даний інтерфейс забезпечує невелику затримку при передачі даних (до 40 мс) і також мінімальне споживання енергії, що забезпечує автономність пристрою. Послідовність

обробки сигналу від смарт-годинника включає фільтрацію та виявлення піків сигналу, а також виділення ознак серцевих скорочень. Дану послідовність розглянуто у підрозділі 2.3.

Процес персоніфікованої адаптації включає послідовне виконання трьох методів, представлених у підрозділах 3.1–3.3. Цей процес формує вхідні дані для методу верифікації, представленою у підрозділі 3.4.

Периферійний сервер використовує FastAPI, щоб прийняти HTTP-запит від користувача, виконати модулі, що реалізують три наведених методи, та повернути рекомендований навчальний модуль. Ці три модулі виконуються як єдиний обчислювальний процес на периферійному сервері, тобто виклики модулів відбуваються безпосередньо в оперативній пам'яті. Така взаємодія на практиці забезпечує затримку в роботі модулів на рівні мікросекунд, що зменшує затримку всього процесу персоніфікованої адаптації. Ключовий фрагмент обробника для даного процесу наведено на рис. 4.2.

```
@app.post("/adapt", response_model=AdaptationResult)
@measure_latency("full_cycle")
async def adaptation_endpoint(event: LearnerEvent, request: Request):
    client = request.app.state.http_client

    # Крок 1 (Метод з 3.1): V_strat – рівень підготовки, профіль помилок
    v_strat = await run_cat(client, event)

    # Крок 2 (Метод з 3.2): V_tact – ЧСС, ВСР, рівень навантаження
    v_tact = await run_lstm(client, event, v_strat)

    # Крок 3 (Метод з 3.3): M* – вибір модуля за V_user = [V_strat, V_tact]
    result = await run_mapc(client, event, v_strat, v_tact)

    # Асинхронна публікація до Kafka для Методу 3.4
    await kafka_producer.send("session.events", {
        "learner_id": event.learner_id,
        "difficulty_level": v_strat["difficulty_level"],
        "theta": v_strat["theta"],
        "cog_load": v_tact["cog_load"],
        "module_selected": result["module"],
        "timestamp": time.time()
    })

    return AdaptationResult(**result)
```

Рисунок 4.3 – Послідовність персоніфікованої адаптації процесу навчання

Виклик платформи `kafka_producer.send(...)` виконується асинхронно. Периферійний сервер спочатку публікує подію в Kafka і тут же повертає відповідь здобувачеві. Сервер не очікує підтвердження від мікросервісного рівня, що забезпечує зменшення затримок у процесі персоніфікованої адаптації процесу вибору навчальних модулів.

Дана архітектура включає ряд програмних компонентів з відкритим вихідним кодом та реалізує три патерни взаємодії в розробленій гібридній архітектурі:

- обробка модулів критичного шляху всередині одного процесу на периферійному рівні;
- синхронна мережева взаємодія між рівнем клієнта і периферійним рівнем, призначена для передачі векторів профілю та стану користувача для подальшого персоніфікованого вибору навчальних модулів;
- асинхронний обмін подіями за допомогою Apache Kafka для взаємодії з мікросервісним рівнем.

Така реалізація гібридної архітектури забезпечує ізоляцію підмереж периферійного рівня `edge_net` й мікросервісного рівня `micro_net` та зменшення затримок для користувачів.

Перевірка затримок забезпечується використанням інструментів з відкритим кодом Jaeger для розподіленого трасування, Prometheus для моніторингу Grafana для інтерактивної візуалізації.

Використання контейнерів та декларативне описання конфігурації для Docker Compose забезпечує відтворюваність експериментальної перевірки для методів, наведених у підрозділах 3.1–3.4.

Також процес збору та обробки біометричного сигналу від фотоплетизмографічного сенсора смарт-годинника є замінюваною складовою. Заміна типу сенсора не приводить до зміни конвеєру персоніфікованої адаптації процесу навчання у програмній системі.

Таким чином, експериментальна перевірка підтвердила можливість реалізації запропонованої гібридної архітектури програної навчальної системи.

4.2 Експериментальна перевірка методів формування векторів профілю та стану користувача

Експериментальна перевірка методу формування вектору профілю користувача виконана для програмної навчальної системи, що використовує гібридну трирівневу архітектуру. На клієнтському рівні фіксуються дії курсанта автошколи при навчанні та біометрична інформація про його стан. На периферійному рівні приймаються рішення про вибір навчальних сценаріїв у м'якому реальному часі. На мікросервісному рівні виконується верифікація процесу персоніфікованої адаптації у програмній навчальній системі.

Метод формування вектору профілю користувача реалізований на периферійному рівні гібридної архітектури, що дозволяє зменшити затримки на персоніфіковану адаптацію процесу навчання.

При реалізації методу враховано особливості стандартизованого датасету Virtual Driving Test [175]. Даний датасет містить записи щодо результатів навчання 4308 водіїв-початківців, що пройшли тест з теоретичних знань щодо водіння, а потім складали дорожній іспит. Датасет містить для кожного водія-початківця дані щодо керування транспортним засобом (швидкість, керування траєкторією автомобіля, гальмування, а також позиція в смузі руху). Дані зібрані з частотою 10 Hz та містять структуровану розмітку дорожніх ситуацій, включаючи перехрестя, небезпечні ділянки, зони біля шкіл та аварійні ситуації, коли один автомобіль пошкоджує задню частину іншого авто.

При проведенні експериментального дослідження враховано, що з 4308 учасників тесту 1096 (близько 25%) не змогли скласти дорожній іспит. Такий розподіл даних використано для оцінки переваг використання профілю користувача навчальної програмної системи.

При використанні датасету Virtual Driving Test (VDT) окремі зони сценаріїв розглядаються як навчальні модулі, вибір яких виконується з використанням модулю диспетчеризації в гібридній програмній архітектурі. Вибір виконується з використанням профілю курсанта автошколи. Це дозволяє

персоніфікувати послідовність навчальних сценаріїв (модулів) для підготовки курсанта автошколи.

Метод формування вектору профілю користувача використовує дані датасету на структурований профіль, придатний для підтримки прийняття рішень. На рис.4.4 наведено приклад вектору профілю користувача у форматі JSON.

```
{
  "user_id": "learner_0042",
  "timestamp": "2024-05-15T10:32:00Z",
  "B_user": -0.42,
  "L_m": "L2",
  "S_profile": {
    "intersection": "attempted",
    "school_zone": "not_attempted",
    "highway_merge": "not_attempted",
    "hazard_reaction": "attempted"
  },
  "U_err": {
    "tactical": 0.47,
    "strategic": 0.21,
    "situational": 0.32
  },
  "Theta_trend": -0.08,
  "vdt_difficulty_agg": 1.85,
  "L_m_vdt": "L2",
  "ore_risk": 0.61
}
```

Рисунок 4.4 – Вектор профілю користувача

JSON – реалізація вектору профілю користувача базується на формальному описі вектору V_{user}^1 у вигляді (3.12). Додатково до складу JSON – файлу при проведенні експерименту включено технічну інформацію: ідентифікатор користувача; мітку часу; точність оцінювання здібностей користувача; типи опрацьованих навчальних сценаріїв тощо.

Параметр `user_id` задає ідентифікатор користувача в навчальній програмній системі. Цей параметр задається у символьному вигляді і

призначений для інтеграції всіх даних про користувача, наприклад журналів сесій, біометричних даних, результатів іспитів.

Параметр timestamp визначає момент формування вектору V_{user}^1 у форматі виду «РІК-МІСЯЦЬ-ДЕНЬЧАС:хвилини:секунди». Даний параметр дозволяє використовувати лише актуальні значення вектору профілю користувача при диспетчеризації навчальних модулів.

Параметр B_{user} є оцінкою латентних здібностей (3.11), що відображають рівень підготовки курсанта автошколи у момент, визначений значенням timestamp. Дана оцінка отримана основі рамках теорії відгуку на завдання IRT. Нульове значення відповідає середньому рівню підготовки курсантів у вибірці. Від’ємні значення відображають рівень нижче, а додатні – вище середнього.

Параметр Θ_{trend} визначає тренд розвитку здібності B_{user} у часі та задається дійсним числом, що відповідно до (3.11) відображає нахил апроксимуючої прямої над рядом оцінок (3.10). Якщо $\Theta_{trend} > 0$, то рівень здібностей користувача підвищується з часом, якщо $\Theta_{trend} < 0$, то підготовка користувача навчальної програмної системи погіршується. Близькі до нуля значення відповідають приблизно однаковому рівню підготовки. Θ_{trend} показує, чи можна підвищувати складність завдань для користувача навчальної програмної системи.

Параметр L_m задає рівень складності завдання (від L_1 до L_5 за (3.7)). Даний рівень використовується модулем диспетчеризації при виборі рівня складності наступних сценаріїв.

Параметр $S_{profile}$ представляє собою опис вкладеного об’єкту, що описує профіль опрацьованих користувачем навчальних сценаріїв; його ключі відповідають типам сценаріїв навчання, наприклад, такі типи як «intersection», «highway_merge» тощо. Значення цих ключів відображають ступінь опанування кожного виду навчального сценарію.

Параметр U_{err} містить частки помилок за (3.9) для тактичного, стратегічного та ситуаційного рівня. Складова $U_{err}(\text{«tactical»})$ відображає частку помилок у тактичних завданнях з керування автомобілем у навчальній

дорожній ситуації. Складова $U_err(\text{«strategic»})$ містить частку стратегічних помилок щодо, наприклад, планування маршруту або вибору траєкторії руху. Складова $U_err(\text{«situational»})$ відображає помилки у ситуації в цілому. Ці дані призначені для вибору навчальних сценаріїв для подальшого вивчення.

Параметр $vdt_difficulty_agg$ представляє усереднену оцінку складності опанованих сценаріїв із Virtual Driving Test. Параметр L_m_vdt задає рівень складності для експериментального датасету VDT. Цей параметр призначений для порівняння узгодженості з L_m . Узгодженість обох показників свідчить про відповідність тестового профілю та реальної поведінки.

Параметр ore_risk є оцінкою ризику невдачі при складанні дорожнього іспиту ORE. Значення, близькі до 1, свідчать про високий ризик.

У представленому векторі значення $B_user = -0,42$ означає рівень підготовки «нижче середнього». Параметр $L_m = L2$ задає базовий рівень складності для вибору навчальних сценаріїв. У групі параметрів U_err переважають тактичні помилки ("tactical": 0,47). Це означає, що мають бути вибрані модулі навчання тактичним маневрам. Від'ємне значення $\Theta_trend = -0,08$ свідчить про незначне погіршення результатів, можливо внаслідок втоми курсанта.

Вектор будується в такій послідовності. Оцінка B_user розраховується таким чином, щоб отримати найбільшу ймовірність відповідей із датасету. На основі B_user визначається рівень складності L_m . Набір помилок U_err розраховується через обчислення частоти помилок для тактичних, стратегічних, ситуативних завдань. Цей набір дає можливість виявити слабкі місця у підготовці користувача. Тренд Θ_trend обчислюється як коефіцієнт нахилу прямої для послідовності оцінок B_user і відображає динаміку навчання користувача.

Додатково, на базі даних із датасету VDT, обчислюється рівень складності опанованих сценаріїв L_m_vdt . Співставлення значень L_m та L_m_vdt дозволяє перевірити відповідність між розрахованим і фактичним рівнями підготовки й в

подальшому використовувати цю інформацію в при диспетчеризації навчальних модулів.

При проведенні експериментальної перевірки виконано порівняння результатів відповідей на тести без урахування параметрів цих тестів, інформативності, складності, а також ймовірності вгадування (Варіант А), результатів відповідей з частковим врахуванням профілю користувача, коли враховано розподіл помилок користувача за типами завдань (Варіант Б), а також результатів при використанні повного вектору профілю користувача.

При проведенні перевірки використано показники AUC, MSE та Acc_L. Результати перевірки наведено в табл. 4.2.

Таблиця 4.2 – Результати експериментальної перевірки методу формування вектору профілю користувача

Підхід	AUC	MSE	Acc_L
Варіант А	0,73	0,87	0,62
Варіант Б	0,76	0,82	0,67
Метод формування вектору профілю користувача навчальної програмної системи	0,83	0,63	0,80

Показник AUC визначається через площу під ROC-кривою для класифікатора. Даний показник на основі оцінки B_{user} визначає, чи зможе користувач скласти іспит ORE. AUC можна в даному випадку інтерпретувати як ймовірність того, що довільний користувач із зданим ORE отримає вищий бал, ніж випадковий користувач із незданим іспитом. Вищі значення показника AUC свідчать про краще розрізнення користувачів, які складають ORE, та користувачів, які не склали цей іспит.

MSE відображає відхилення між передбаченням ймовірності успішного складання ORE на основі оцінки B_{user} та фактичним результатом ORE у датасеті. MSE обчислюється як середнє значення різниці квадратів між прогнозованим та фактичним результатом курсанта.

Показник Acc_L відображає ступінь збігу рівнів складності L_m та L_m_vdt . Перший отриманий на основі оцінки B_user , а другий міститься в датасеті VDT.

Наведені у таблиці 4.2 порівняльні оцінки показують покращення всіх показників при реалізації розробленого методу. Збільшення AUC та зменшення MSE при використанні методу побудови профілю користувача свідчить про підвищення можливості розрізняти тих користувачів-курсантів, які складуть іспит, а також тих, хто не зможе скласти. Підвищення Acc_L означає, що отримані при виконанні методу рівні складності L достатньо узгоджені із фактичними рівнями сценаріїв у датасеті Virtual Driving Test. Таким чином, використання розробленого методу формування вектору профілю користувача дає можливість отримати оцінку рівня знань та здібностей користувача, що відповідає експериментальним даним.

Метод побудови вектору поточного стану користувача НПС визначає характеристику курсанта автошколи в іншому масштабі часу, задаючи опис характеристик користувача під час поточної навчальної сесії, та враховує нормалізовану частоту серцевих скорочень, варіабельність серцевого ритму, а також рівень когнітивного навантаження $Cognitive_user$. Останній класифікується за значеннями із множини $\{LOW, OPTIMAL, HIGH\}$.

Використання другого вектору при диспетчеризації дає можливість прийняти коректне рішення щодо вибору навчального сценарію. Курсант із рівнем підготовки вище середнього та станом когнітивного навантаження HIGH має отримати спрощений початковий сценарій, незалежно від високого рівня його підготовки. Той же самий курсант зі станом когнітивного навантаження OPTIMAL може опановувати складніший навчальний модуль. Проте існуючі адаптивні навчальні системи, такі як Intelligent Tutoring Systems та LMS, використовують статичну оцінку рівня підготовки і не враховують зміни когнітивного навантаження користувача, що може привести до помилок при диспетчеризації навчальних модулів.

Метод виконує послідовність обробки біометричних сигналів у режимі м'якого реального часу із затримкою менше 1 с. від отримання RR-інтервалу до побудови оновленого вектору стану користувача.

Варіабельність серцевого ритму є індикатором балансу між симпатичною (активується в стресовій ситуації) та парасимпатичною (активується у спокійному стані) частинами автономної або вегетативної нервової системи. Для стану LOW когнітивного навантаження характерна значна варіабельність RR-інтервалів (в діапазоні 30–50 мс) внаслідок парасимпатичної активності. При високому (HIGH) рівні когнітивного навантаження активною є , типовий діапазон становить 30–50 мс симпатична активність приводить до стабілізації RR-інтервалів і зниження варіабельності до 5–15 мс. Такі патерни виявляються за допомогою LSTM-класифікатору.

Побудова вектору поточного стану користувача реалізується у такій послідовності. На першому етапі виконується збір RR-інтервалів від смарт-годинника через Bluetooth Low Energy. Використовується кільцевий буфер із 180 елементів, щоб зберегти не менше останніх 90 секунд вхідного сигналу.

На другому етапі виконується фільтрація вхідних даних. RR-інтервал помічається як аномалія за такими показниками:

- за пороговим значенням $RR < 300$ мс або $RR > 2000$ мс;
- за відносною різницею зміною між сусідніми інтервалами більше ніж 20%;
- за відхиленням від локальної медіани (для поточного вікна), коли відхилення становить більше трьох значень медіани абсолютних відхилень.

Аномальні значення замінюються на медіанне значення чотирьох сусідніх нормальних інтервалів. Медіанна фільтрація замість середнього використовується тому що медіана не залежить від викидів.

На третьому та четвертому етапах розраховується середньоквадратичне відхилення HRV та інтегральний показник стану серця H – ознаки.

На етапі 5 виконується LSTM-класифікація поточного стану користувача. Послідовність даних для 10 вікон F_HRV подається у LSTM – класифікатор із softmax на виході. Реалізацію трьохшарового класифікатора наведено на рис. 4.5.

```
class CogLoadLSTM(nn.Module):
    def __init__(self):
        super().__init__()
        self.lstm1 = nn.LSTM(6, 64, batch_first=True)
        self.lstm2 = nn.LSTM(64, 32, batch_first=True)
        self.fc = nn.Linear(32, 3)
```

Рисунок 4.5 – Реалізація LSTM – класифікатора

У класифікаторі шар lstm1 містить 64 нейрони і виявляє базові залежності для вхідних 10 вікон. Шар lstm2 виявляє складні паттерни на основі базових залежностей. fc – шар видає 3 логіти, по одному для класу LOW, OPTIMAL та HIGH. Логіти виходять за межі діапазону $[0,1]$.

Використання softmax в подальшому дає можливість на основі логітів отримати розподіл ймовірностей p_LOW , $p_OPTIMAL$, p_HIGH по класах когнітивного навантаження. Cognitive_user визначається для класу із максимальним значенням ймовірності. Якщо отримане найбільше значення ймовірності менш ніж 0,65, то стан вважається незмінним і у Cognitive_user залишається попереднє значення.

На етапі 6 формується вектор поточного стану користувача, що включає основні дані про стан користувача HRV_norm, HR_norm, Cognitive_user, мітку часу, а також проміжні дані, що призначені для уточнення когнітивного навантаження.

На рис.4.6 наведено приклад вектору стану користувача, представлений у форматі JSON.

У даному прикладі $HR_norm = 0,30$ означає перевищення на 30% базового рівня спокою. $HRV_norm = 15,3$ мс є значно нижчою за норму спокою, що вказує на активацію симпатичної активності. cognitive_user = “HIGH” зі значенням

$p_high = 0,89$ означає, що диспетчер навчальних модулів має знизити складність наступного навчального сценарію.

```
{
  "user_id": "learner_0042",
  "timestamp": 1732290000000,
  "hr_norm": 0.30,
  "hrv_norm": 15.3,
  "cognitive_user": "HIGH",
  "p_low": 0.06,
  "p_optimal": 0.05,
  "p_high": 0.89,
}
```

Рисунок 4.6 – Вектор стану користувача

Експериментальну перевірку методу виконано на датасеті WESAD, який відображає виявлення стресу з використанням носимих сенсорів [176]. В датасеті відображено використання нагрудного і наручного пристроїв. Наручний пристрій відображає пульс людини. Датасет описує три стани (нейтральний, стрес, розвага), що відповідають станам LOW, OPTIMAL, HIGH у розробленому методі.

При проведенні експериментальної перевірки методу використано показники Ассурасу та F1-міра для класу HIGH (F1_HIGH). Ассурасу (точність) показує частку правильних прогнозів від числа усіх прогнозів. F1-міра для класу HIGH є збалансованою оцінкою здатності виявити високе когнітивне навантаження, яка враховує точність та повноту. Крім точності вибрано F1-міру тому що точність може бути високою якщо клас HIGH є рідкісним і він погано виявляється.

Результати перевірки методу побудови вектору поточного стану користувача наведено в табл. 4.3.

Результати експериментальної перевірки показують, що використання LSTM в розробленому методі дає вигравш в точності до 8% за рахунок виявлення трендів стану користувача, наприклад наростання когнітивного навантаження.

Таблиця 4.3 – Результати експериментальної перевірки методу формування вектору поточного фізіологічного стану користувача

Підхід	Accuracy	F1_HIGH
SVM, аналіз поточних даних	0,79	0,72
Random Forest, 5 послідовних вікон	0,81	0,76
Метод побудови вектору стану користувача навчальної програмної системи, 10 послідовних вікон	0,87	0,84

Ці тренди виявляються за рахунок порівняння з попередніми станами та згладжування викидів. Проте витрати часу є найбільшими і для розробленого методу становлять близько 0,2 секунди, що на практиці є достатнім для диспетчеризації навчальних модулів.

4.3 Експериментальна перевірка методів персоніфікованої адаптації та верифікації процесу вибору навчальних модулів

Метод персоніфікованої адаптації навчальних модулів реалізований у модулі-диспетчері навчальної програмної системи, виконує вибір наступного сценарію на основі виконання базових етапів аналізу стану, формування пріоритетного списку кандидатів та виконання рішення.

Диспетчер в гібридній програмній архітектурі підписується на два потоки подій:

- оновлення вектору профілю користувача після навчальної сесії в програмній системі;
- оновлення вектору стану користувача, яке відбувається кожну хвилину активної сесії навчання.

Обидва вектори розміщені у спільній структурі на периферійному рівні та зчитуються модулем диспетчера з мінімальною затримкою.

Структура рішення модуля-диспетчера за результатами виконання методу у форматі JSON представлена на рис. 4.7.

```

{
  "decision_id": "dd_0042_0017",
  "user_id": "learner_0042",
  "timestamp": 1732291200000,
  "v_profile_snapshot": {
    "B_user": -0.42,
    "L_m": "L2",
    "U_err": {"tactical": 0.47, "strategic": 0.21, "situational": 0.32},
    "Theta_trend": -0.08
  },
  "v_state_snapshot": {
    "cognitive_user": "HIGH",
    "p_low": 0.06,
    "p_optimal": 0.05,
    "p_high": 0.89
  },
  "d_target": "L1",
  "t_target": "tactical",
  "candidates": [
    {"module_id": "intersection_L1_v3", "rank_score": 0.91},
    {"module_id": "lane_change_L1_v2", "rank_score": 0.84},
    {"module_id": "hazard_reaction_L1_v1", "rank_score": 0.76}
  ],
  "selected_module": "intersection_L1_v3",
  "dispatch_rationale": "HIGH_load_downgrade + tactical_err_dominant",
  "fallback_used": false,
  "latency_ms": 38
}

```

Рисунок 4.7 – Структура рішення за результатами виконання методу персонафікованої адаптації процесу навчання

У наведеному рішенні курсант автошколи має рівень L2, проте поточний стан когнітивного навантаження є HIGH приводить до зниження D_target з L2 до рівня L1. Користувач навчальної системи допускає переважно тактичні помилки, U_err{tactical: 0,47}, що приводить до значення T_target = "tactical". Запропонований навчальний модуль intersection_L1_v3 є сценарієм маневру початкового рівня на регульованому перехресті.

Поле dispatch_rationale містить інформацію для подальшої верифікації процесу персонафікованої адаптації з використанням process mining на мікросервісному рівні. Затримка на прийняття рішення становить 38 мс, що забезпечує роботу в режимі м'якого реального часу.

Експериментальна перевірка методу виконана на датасеті VDT. Кожен запис датасету VDT розбивається на зони подій, що відображають сценарії у навчальних модулях. Показники когнітивного навантаження розраховано на основі дослідження [29], через відстань відхилень від центру дороги, кількість корекцій, частоту гальмування. Велика кількість корекцій руху у датасеті свідчить про високе когнітивне навантаження, а невеликі відхилення із незначними корекціями відображають низьке навантаження.

При проведенні перевірки порівнюються три варіанти вибору сценаріїв:

- стандартний навчальний план, що містить фіксований порядок навчальних сценаріїв;
- адаптований план навчання за значенням параметрів, що включені до профілю користувача;
- вибір навчальних сценаріїв за методом персоніфікованої адаптації процесу навчання.

Результати експериментальної перевірки методу персоніфікованої адаптації наведено у табл. 4.4

Кожна конфігурація навчальних сценаріїв оцінювалась за трьома показниками, які вимірюють різні характеристики рішень щодо вибору наступного сценарію. Частка невідповідних до рівня користувача призначень відображає рішення, коли новий модуль був занадто складним для користувача-новачка або занадто простим для досвідченого курсанта. Частку рішень, що підтримують розвиток знань і навичок користувача, включає сценарії навчання з дещо більш високою складністю порівняно з його поточним рівнем підготовки. Частка запропонованих рішень із перевантаженням не враховує поточний рівень когнітивного перенавантаження користувача.

Базова конфігурація зі стандартним планом навчання демонструє найнижчі показники після виконання диспетчеризації. Використання профілю користувача навчальної програмної системи забезпечує суттєве підвищення показників. Частка рішень, що не відповідає рівню підготовки користувача, знижується до 0,22, частка рішень, що забезпечують ефективне навчання, зростає до 0,58, частка

проблемних рішень з перевантаженням користувача знижується до 0,24. Ці результати підтверджують суттєвий вплив інформації щодо рівня знань користувача на якість адаптації. Сума значень цих показників не дорівнює одиниці, тому що вони вимірюють різні характеристики рішення. Експериментальні дослідження ефективності методу персоніфікованої адаптації процесу навчання засновані на порівнянні результатів застосування розробленого методу із результатами, що демонструють стандартні засоби організації навчального процесу.

Таблиця 4.4 – Результати експериментальної перевірки методу персоніфікованої адаптації процесу навчання

Конфігурація навчальних сценаріїв	Частка рішень, що не відповідають рівню підготовки користувача НПС (занадто легке або занадто складне)	Частка рішень, що забезпечують ефективне навчання	Частка рішень, що не враховують стан HIGH когнітивного навантаження
Стандартний навчальний план з фіксованим порядком навчальних сценаріїв	0,34	0,22	0,31
Адаптований план навчання з використанням вектору профілю користувача;	0,22	0,58	0,24
Метод персоніфікованої адаптації процесу навчання	0,09	0,74	0,07

Після використання розробленого методу персоніфікованої адаптації частка рішень, що не відповідає рівню підготовки користувача, знижується до лише 0,09, суттєво, до 0,74 зростає частка рішень, що забезпечують ефективне навчання, та зменшується можливість перенавантаження користувача до 0,07. Таким чином, розроблений метод досягає найкращих значень за всіма показниками і забезпечує підвищення значень показників у декілька разів.

Результати методу персоніфікованої адаптації перевіряються з використанням методу верифікації, який реалізує побудову еталонного процесу, зіставлення з ним фактичних траєкторій із логу та виявлення порушень. При проведенні експерименту використано датасет VDT. Кожен запис датасету розбивався на дорожні ситуації відповідно до оригінальних міток. Такі ситуації включають рух на перехресті, на небезпечних ділянках тощо. Для кожної зони обчислювались ключові ознаки: швидкість, частота гальмувань, середнє та максимальне відхилення від смуги руху. На основі отриманих даних формувались вектори профілю та стану користувача, як було розглянуто в підрозділі 4.2. Для кожної ситуації та кожного водія дані упорядковувались у траси, що відображали траєкторію опанування навчальних сценаріїв. Кожна траса містить послідовність подій проходження дорожньої ситуації.

Приклад опису події при опануванні дорожньої ситуації наведено на рис.4.8.

```
<event>
  <string key="concept:name"      value="intersection_traffic_light"/>
  <date key="time:timestamp"      value="2024-05-15T09:00:00.000+02:00"/>
  <string key="action"            value="CONNECT"/>
  <string key="module_difficulty" value="L1"/>
  <string key="cognitive_userload" value="OPTIMAL"/>
  <float key="lane_offset_avg"    value="0.18"/>
  <string key="violation_type"    value="NONE"/>
  <string key="violation_critical" value="false"/>
</event>
```

Рисунок 4.8 – Структура події для сценарію дорожньої ситуації

Подія надана у форматі хес, призначеному для опису процесних траєкторій.

Опис події має таку структуру. Змінна `<string key="concept:name" value="intersection_traffic_light"/>` містить назву навчального сценарію «перехрестя зі світлофором». Змінна `<date key="time:timestamp" value="2024-05-15T09:00:00.000+02:00"/>` є міткою часу проїзду через перехрестя зі

світлофором. Змінна `<string key="action" value="CONNECT"/>` містить рішення модуля диспетчеризації щодо підключення даного дорожнього сценарію користувачеві. Змінна `<string key="module_difficulty" value="L1"/>` містить рівень складності сценарію дорожньої ситуації. У змінній `<string key="cognitive_userload" value="OPTIMAL"/>` наведено рівень когнітивного навантаження курсанта автошколи у момент опанування дорожньої ситуації. Значення `<float key="lane_offset_avg" value="0.18"/>` відображає відхилення при проїзді за даним сценарієм. Це значення міститься у датасеті і може бути використано для перевірки `cognitive_userload` та інтерпретації порушень. Тип виявленого порушення наведений у `<string key="violation_type" value="NONE"/>`. Значення `NONE` означає відсутність порушення. Рішення модуля диспетчеризації відповідає цільовій моделі процесу персоніфікованої адаптації. Інформація про порушення у двійковому форматі міститься у змінній `<string key="violation_critical" value="false"/>`. Значення `false` показує, що порушення відсутнє. Дана змінна використовується для визначення кількості порушень при проведенні експерименту.

Еталонні процеси навчання визначались за умови відповідності рівня підготовки користувача вимогам навчального сценарію та відсутності високого (HIGH) когнітивного навантаження.

Вимоги до складності ситуацій встановлені з урахуванням розмітки датасету. Наприклад, дорожнім ситуаціям з мінімальними навичками (водіння на рівній ділянці тощо) відповідали мінімальні рівні вимог до підготовки та найвищий рівень когнітивного навантаження, а ситуаціям виду водіння на нічній трасі, в дощову погоду – найвищі рівні вимог до підготовки та найнижчий рівень когнітивного навантаження.

При перевірці методу верифікацію використовувались показники `Fitness` та `Precision`, розраховувались частки порушень правил персоніфікованої адаптації.

Показник `Fitness` відображає частку трас при виконанні персоніфікованої адаптації, яка повністю відповідає цільовій моделі як еталонному опису

дорожньої ситуації. Ці реалізовані траєкторії навчання не мають протиріч із правилами персоніфікованої адаптації.

Показник Precision вимірює частку подій на траєкторії навчання, що були реально реалізовані при адаптації. Високе значення Precision показує, що модель описує реальний процес персоніфікованої адаптації та не містить таких варіантів, яких не існує при реальному навчанні.

При перевірці порушень враховувались завищена складність сценарію (порушення адаптації), та стан HIGH когнітивного навантаження (порушення персоніфікації). Показник частки порушень персоніфікованої адаптації враховує випадки невиконання будь-якої з цих умов. Показник частки критичних порушень враховує лише ті рішення, де порушуються обидві умови.

Розрахунок показників виконувався для зростаючих підвбірок журналів (перші 100, 500, 1000 трас та повний датасет) з метою порівняння значень отриманих показників.

Результати експериментальної перевірки методу персоніфікованої адаптації наведено у табл. 4.5.

Таблиця 4.5 – Результати експериментальної перевірки методу персоніфікованої адаптації процесу навчання

Вибірка трас	Fitness	Precision	Частка порушень персоніфікованої адаптації	Частка критичних порушень персоніфікованої адаптації
100 трас	0,86	0,77	0,12	0,034
500 трас	0,87	0,78	0,11	0,031
1000 трас	0,87	0,78	0,11	0,033
Повний датасет	0,87	0,78	0,11	0,032

Значення показника Fitness на більшості вибірок становить 0,87, що підтверджує, що переважна більшість виконаних траєкторій проходження навчальних сценаріїв відповідає процесу персоніфікованої адаптації. Значення показника Precision 0,78 означає, що правила адаптації не містять значної

кількості теоретично можливих, проте не реалізованих на практиці переходів до нових навчальних модулів.

Частка порушень одного з правил персоніфікованої адаптації у 11% показує, що метод верифікації може виявляти рішення, які мають бути уточнені при удосконаленні процесу персоніфікованої адаптації навчання. Значення частки критичних порушень показує, що 3,2% відсотка рішень містять некоректну адаптацію, яка може привести до когнітивного перевантаження та подальшого формування негативних шаблонів водіння у складних дорожніх сценаріях.

Динаміка значень показників на вибірках трас свідчить, що метод верифікації можна використовувати для журналів невеликого обсягу.

Структура порушень в залежності від персоніфікації або адаптації наведена у табл. 4.6.

Таблиця 4.6 – Структура порушень при адаптації на персоніфікації процесу навчання

Тип порушення при персоніфікованій адаптації процесу навчання	Опис	Частка від всіх порушень
Адаптивний	Складність сценарію перевищує рівень підготовленості курсанта автошколи	0,41
Персоніфікований	Ігнорування підвищеного стресу або HIGH навантаження	0,38
Критичний	Одночасне порушення обох умов	0,21

Більшість порушень є адаптивними, що свідчить про невідповідність рівня складності навчального сценарію та поточного рівня знань та навичок користувача. Ці дані підтверджують результати у табл. 4.4 зі значною кількістю порушень для стандартного навчального плану із фіксованих порядком

навчальних сценаріїв. Велика частка порушень персоніфікації свідчить про окремий вплив даного компонента, коли вплив когнітивного перенавантаження не компенсується інформацією із профілю користувача. Критичні порушення з часткою 21% мають найбільший вплив на ефективність навчання. Сценарії з такими порушеннями мають бути відкориговані в першу чергу.

Висновки до розділу 4

1. Виконано реалізацію та експериментальну перевірку гібридної архітектури навчальної програмної системи автошколи. Реалізована архітектура використовує три патерни взаємодії, а саме обробку модулів критичного шляху в одному процесі на периферійному рівні, синхронну мережеву взаємодію між рівнем клієнта і периферійним рівнем, асинхронний обмін подіями для взаємодії з мікросервісним рівнем. Така реалізація розробленої архітектури забезпечує ізоляцію підмереж периферійного рівня та мікросервісного рівня та відповідне зменшення затримок для користувачів.

2. Проведено експериментальну перевірку методів формування векторів профілю та стану користувача навчальної програмної системи. Метод формування вектору профілю користувача забезпечив узгодженість отриманої оцінки рівня підготовки користувача з фактичними рівнями складності навчальних сценаріїв у експериментальному датасеті. Метод побудови вектору поточного стану користувача забезпечив можливість більш точного визначення когнітивного навантаження за рахунок виявлення трендів стану користувача.

3. Проведено експериментальну перевірку методів персоніфікованої адаптації та верифікації процесу навчання. Метод персоніфікованої адаптації забезпечив досягнення найкращих значень за показниками частки рішень, що не відповідають рівню підготовки користувача, частки рішень, що забезпечують ефективне навчання, а також частки рішень, що не враховують когнітивне перенавантаження користувача. Метод верифікації забезпечив можливість удосконалення реального процесу персоніфікованої адаптації на основі

виявлення критичних порушень, пов'язаних із невідповідністю між навчальними сценаріями та рівнем підготовки користувача, а також із перевищенням рівня когнітивного навантаження.

ВИСНОВКИ

У дисертаційній роботі представлено нові наукові та практичні результати при вирішенні актуального наукового завдання розробки моделі та методів персоніфікованої адаптації в програмних системах навчання в автошколі для підвищення ефективності підготовки водіїв з урахуванням як рівня підготовки користувача, так і його поточного когнітивного стану. Результати впровадження методів персоніфікованої адаптації свідчать про перспективність використання розроблених методичних засобів у навчальний процес інших типів закладів.

В ході дослідження отримано такі результати.

1. Аналіз адаптивного процесу навчання водіїв та підходів до персоніфікації навчання в автошколах показав, що ефективність підготовки водіїв значною мірою залежить від контролю когнітивного навантаження користувачів навчальних систем, що потребує виконання персоніфікованої адаптації процесу навчання. Сучасні підходи до персоніфікації враховують стан користувача та не враховують рівень його підготовки, а підходи до адаптації враховують лише рівень підготовки користувача. Дане обмеження пов'язано із реалізованими архітектурними рішенням навчальних програмних систем. Реалізація комплексного підходу до персоніфікованої адаптації процесу навчання потребує розробки гібридної програмної архітектури, що поєднує можливості масштабованості мікросервісної архітектури для удосконалення процесу навчання та швидкої персоніфікованої адаптації на базі монолітної архітектури.

2. Вперше запропоновано модель гібридної програмної архітектури з периферійними обчисленнями системи навчання в автошколі. Модель включає клієнтський, периферійний та мікросервісний рівні. Клієнтський рівень архітектури забезпечує візуалізацію дорожнього середовища та отримання біометричних даних користувача. На периферійному рівні виконується персоніфікована адаптація процесу навчання на основі обробки біометричних даних та вектору профілю користувача. Мікросервісний рівень забезпечує

верифікацію процесу персоніфікованої адаптації. Розроблена модель забезпечує можливість персоніфікованої адаптації процесу навчання у програмних системах автошкіл на основі комплексного урахування рівня знань та поточного рівня когнітивного навантаження користувача.

3. Удосконалено метод побудови вектору профілю користувача навчальної програмної системи, який на відміну від існуючих виконує оцінку латентних здібностей користувача з використанням психометричних моделей, визначає на цій основі рівні складності задач для користувача, опис типових помилок і формує вектор профілю користувача як формальну специфікацію інтерфейсу з користувачем. Метод дає можливість автоматизувати персоніфіковану адаптацію та верифікацію модулів навчальної програмної системи з урахуванням значень компонентів вектору профілю користувача.

4. Удосконалено метод побудови вектору поточного стану користувача навчальної програмної системи, який на відміну від існуючих виконує аналіз варіабельності серцевого ритму, визначає патерни серцевого ритму та формує вектор поточного стану користувача, що може бути представлений в рамках програмного інтерфейсу для автоматизованої диспетчеризації модулів навчальної програмної системи. Метод забезпечує можливість оцінки стану користувача у м'якому реальному часі у гібридній програмній архітектурі адаптивної навчальної системи.

5. Вперше запропоновано метод персоніфікованої адаптації процесу навчання шляхом вибору навчальних модулів на основі векторів профілю та стану користувача. Метод, на відміну від існуючих, враховує рівень підготовки, поточний психофізичний стан користувача, а також ступінь новизни завдань у навчальних модулях і використовує попередню фільтрацію модулів за ступенем відповідності рівню підготовки користувача, що дає можливість скоригувати процес навчання з урахуванням як рівня знань, так і когнітивного навантаження користувача навчальної програмної системи.

6. Набув подальшого розвитку метод верифікації адаптивної програмної системи з використанням технології інтелектуального аналізу процесів. Метод

включає етапи побудови журналу подій адаптивної навчальної програмної системи, побудови моделей цільового та фактичного процесів персоніфікованої адаптації, виявлення порушень адаптивної логіки на базі порівняння цільової та фактичної моделей процесу персоніфікованої адаптації. Метод створює умови для удосконалення процесу персоніфікованої адаптації при навчанні з урахуванням різниці між моделями цільового та фактичного процесу навчання.

7. Виконано реалізацію гібридної програмної архітектури та експериментальну перевірку розроблених методів персоніфікованої адаптації процесу навчання. Реалізована архітектура використовує різні патерни взаємодії на клієнтському, периферійному та мікросервісному рівнях, що забезпечує зменшення затримок при адаптації процесу навчання для користувачів. Експериментальна перевірка методів персоніфікованої адаптації показала суттєве підвищення частки рішень, що забезпечують ефективне навчання та зниження частки рішень, що не враховують когнітивне навантаження користувача порівняно з існуючими підходами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Малюга А. І. Концептуальні засади дослідження впровадження технологій віртуальної реальності у навчальний процес автошкіл // *Вчені записки ТНУ імені В. І. Вернадського. Серія: Технічні науки*. – 2024. – Т. 35 (74), № 1. – С. 214–221. – DOI: 10.32782/2663-5941/2024.1.1/33.

2. Малюга А. І. Гібридна програмна архітектура з периферійними обчисленнями для адаптивних VR-систем навчання водінню з біометричним зворотним зв'язком // *Авіаційно-космічна техніка і технологія*. – 2025. – № 5. – С. 96–111. – DOI: 10.32620/aktt.2025.5.09.

3. Малюга А. І. Методи побудови адаптивного програмного забезпечення навчальних систем // *Авіаційно-космічна техніка і технологія*. – 2025. – № 6. – С. 113–127. – DOI: 10.32620/aktt.2025.6.11.

4. Малюга А. І. Метод онлайн-диспетчеризації тематичних модулів в адаптивних навчальних системах із використанням векторів індивідуального профілю та поточного стану користувача // *Авіаційно-космічна техніка і технологія*. – 2026. – № 1. – С. 123–132. – DOI: 10.32620/aktt.2026.1.11.

5. Малюга А. І., Соколова Є. В. Метод верифікації адаптивних програмних систем з використанням технології process mining // *Відкриті інформаційні та комп'ютерні інтегровані технології*. – 2026. – № 107. – С. 257–263. – DOI: 10.32620/oikit.2026.107.17.

6. Малюга А. І. Шляхи підвищення ефективності процесу навчання навичкам водіння автомобільної та спеціальної техніки за рахунок впровадження спеціалізованих програмних засобів // *Проблеми інформатизації: тези доповідей дванадцятої міжнародної науково-технічної конференції (21–22 листопада 2024 р.)*. – Баку – Харків – Бельсько-Бяла, 2024. – Т. 2, секція 4. – С. 28.

7. Малюга А. І. Особливості застосування штучних нейронних мереж у тренажерах для підготовки водіїв автомобільної та спеціальної техніки // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів*

управління: тези доповідей п'ятнадцятої міжнародної науково-технічної конференції (24–25 квітня 2025 р.). – Баку – Харків – Жиліна, 2025. – Т. 1, секції 1, 5. – С. 28.

8. Малюга А. І. Моделювання процесу обробки біометричних даних у середовищі віртуальної реальності при адаптивному навчанні водіїв // *Проблеми інформатизації: тези доповідей тринадцятої міжнародної науково-технічної конференції (27–28 листопада 2025 р.).* – Баку – Харків – Бельсько-Бяла, 2025. – Т. 1, секції 1, 2. – С. 36.

9. Малюга А. І. Верифікація адаптивних програмних систем, що змінюють поведінку у відповідності до поточного психофізіологічного стану користувача // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей шістнадцятої міжнародної науково-технічної конференції (23–24 квітня 2026 р.).* – Баку – Харків – Жиліна, 2026. – Т. 1, секція 2. – С. 43.

10. Шостак І. В., Малюга А. І. Методичні і програмні засоби персоніфікованої адаптації навчального процесу в автошколі // *Інтегровані інтелектуальні робототехнічні комплекси (ІРТК-2026): матеріали дев'ятнадцятої міжнародної науково-практичної конференції (19–20 травня 2026 р.).* – Київ: НАУ, 2026. – С. 557-558.

11. Постанова Кабінету Міністрів України від 02.03.2010 № 229 "Про затвердження Типової навчальної програми підготовки та перепідготовки водіїв транспортних засобів". URL: <https://zakon.rada.gov.ua/go/229-2010-п> (дата звернення: 04.08.2025).

12. МВС України. (2024). Основні зміни до Типової навчальної програми підготовки та перепідготовки водіїв транспортних засобів. URL: <https://mvs.gov.ua/upload/document/Y13vOyyvOYVhyF4JYb017UXvvbYWHL8XnTay8fw1.docx> (дата звернення: 04.08.2025).

13. Allstate. (2024). California graduated driver licensing laws. URL: <https://www.allstate.com/auto-insurance/california-graduated-driver-licensing-laws> (дата звернення: 04.08.2025).

14. DrivingTheory4All. (2023). Theory test pass marks and pass rates. URL: <https://www.drivingtheory4all.co.uk> (дата звернення: 04.08.2025).

15. European Confederation of Driving Schools (ECF). (2024). Conduite Accompagnée : Fonctionnement du Permis B AAC. URL: <https://www.ecf.asso.fr> (дата звернення: 04.08.2025).

16. En Voiture Simone. (2022). Rendez-vous pédagogiques pour la conduite accompagnée. URL: <https://www.envoituresimone.com/permis-de-conduire/conseils-conduite/formation-conduite/conduite-accompagnee/rendez-vous-pedagogique> (дата звернення: 04.08.2025).

17. Ornika. (2024). Les rendez-vous pédagogiques liés à la conduite accompagnée. URL: <https://www.ornika.com/conduite-accompagnee/aac-realiser-inscription/rendez-vous-pedagogiques> (дата звернення: 04.08.2025).

18. Ministère de l'Éducation nationale. (2021). Conduite accompagnée: lancement d'une campagne d'information auprès des élèves de 4e des départements. URL: <https://www.education.gouv.fr/conduite-accompagnee-lancement-d-une-campagne-d-information-aupres-des-eleves-de-4e-des-departements-450016> (дата звернення: 04.08.2025).

19. GOV.UK. (2016). Theory test: cars: Hazard perception test. URL: <https://www.gov.uk/theory-test/hazard-perception-test> (дата звернення: 04.08.2025).

20. GOV.UK. (2021). Driver and rider testing and instructor statistics: April 2020 to March 2021. URL: <https://www.gov.uk/government/statistics/driver-and-rider-testing-and-instructor-statistics-april-2020-to-march-2021/driver-and-rider-testing-and-instructor-statistics-april-2020-to-march-2021> (дата звернення: 04.08.2025).

21. Korteori. (2025). Risk training part 1 and 2 – Theory. URL: <https://korteori.se> (дата звернення: 04.08.2025).

22. Korkortonline. (2025). Skid training – risk training part 2 for a driving licence. URL: <https://korkortonline.se> (дата звернення: 04.08.2025).

23. Accident Analysis & Prevention. (2014). Driver education research article. URL: <https://www.sciencedirect.com/science/article/abs/pii/S1369847814000795> (дата звернення: 04.08.2025).
24. Transportation Research Part F: Traffic Psychology and Behaviour. (2022). Driver training methodology research. URL: <https://www.sciencedirect.com/science/article/pii/S1369847822002819> (дата звернення: 04.08.2025).
25. McCartt A. T., Shabanova V. I., Leaf W. A. Driving experience, crashes and traffic citations of teenage beginning drivers. *Accident Analysis & Prevention*. 2003. Vol. 35, No. 3. P. 311–320.
26. RAC. (2024). How to pass your hazard perception test. URL: <https://www.rac.co.uk> (дата звернення: 04.08.2025).
27. Головний сервісний центр МВС. (2024, листопад 22). Скільки триває практичне навчання та перепідготовка в автошколі відповідно до обраної категорії. URL: <https://hsc.gov.ua/2024/11/22/skilki-trivaye-praktichne-navchannya-ta-perepidgotovka-v-avtoshkoli-vidpovidno-do-obranoyi-kategoriyi/> (дата звернення: 04.08.2025).
28. Головний сервісний центр МВС. (2024, травень 21). Як отримати посвідчення водія вперше. URL: <https://hsc.gov.ua/2024/05/21/yak-otrimati-posvidchennya-vodiya-vpershe-2/> (дата звернення: 04.08.2025).
29. Engström J., Markkula G., Victor T., Merat N. Effects of cognitive load on driving performance: The cognitive control hypothesis. *Human Factors*. 2017. Vol. 59, No. 5. P. 734–764. URL: https://eprints.whiterose.ac.uk/110554/7/HF_Paper_reviewied_final.pdf (дата звернення: 04.08.2025).
30. Novice driver education advancements research. *arXiv preprint arXiv:2503.05762*. 2025. URL: <https://arxiv.org/pdf/2503.05762.pdf> (дата звернення: 04.08.2025).
31. Strayer D. L., Drews F. A., Johnston W. A. Cell phone-induced failures of visual attention during simulated driving. *Journal of Experimental Psychology*:

Applied. 2003. Vol. 9, No. 1. P. 23–32. URL: <https://pubmed.ncbi.nlm.nih.gov/28186421/> (дата звернення: 04.08.2025).

32. Just-Avto. Правила та зміни в процесі здачі на права водія у 2024 році. URL: <https://just-avto.kiev.ua/pravila-ta-zmini-v-protsesi-zdachi-na-prava-vodiya-u-2024-rotsi/> (дата звернення: 04.08.2025).

33. TC Driving. Як отримати права у 2024 році в Україні? URL: <https://tcdriiving.com.ua/yak-otrimaty-prava-u-2024-roci-v-ukraini/> (дата звернення: 04.08.2025).

34. Anderson J. R. Acquisition of cognitive skill. *Psychological Review*. 1982. Vol. 89, No. 4. P. 369–406. DOI: 10.1037/0033-295X.89.4.369.

35. Fitts P. M., Posner M. I. *Human performance*. Belmont, CA: Brooks/Cole. 1967. – узагальнений сучасний виклад див.: Krakauer J. W., Hadjiosif A. M., Xu J., Wong A. L. The role of strategies in motor learning. *Frontiers in Human Neuroscience*. 2019. Vol. 13. P. 1–18. DOI: 10.3389/fnhum.2019.00345.

36. Transportation Research Part F: Traffic Psychology and Behaviour. (2024). Advanced driver training research. URL: <https://www.sciencedirect.com/science/article/pii/S1369847824003073> (дата звернення: 04.08.2025).

37. Visit Ukraine Today. (2024). How to pass the driving test and get a driving licence in Ukraine in 2024. URL: <https://visitukraine.today/uk/blog/4328/how-to-pass-the-driving-test-and-get-a-driving-licence-in-ukraine-in-2024> (дата звернення: 04.08.2025).

38. *Applied Ergonomics*. (2022). Driver training effectiveness study. URL: <https://www.sciencedirect.com/science/article/abs/pii/S0149718922000593> (дата звернення: 04.08.2025).

39. NaUrok. Вдосконалення водійських навичок. URL: <https://naurok.com.ua/vdoskonalennya-vodiyskih-navichok-273503.html> (дата звернення: 04.08.2025).

40. PMC. Virtual reality applications in driver training and education. URL: <https://pmc.ncbi.nlm.nih.gov/articles/PMC12069486/> (дата звернення: 04.08.2025).

41. DriveTech. Why VR is the future of driver training. URL: <https://www.drivetech.co.uk/why-vr-is-the-future-of-driver-training/> (дата звернення: 04.08.2025).
42. Habitus. (2020). Соціологічні аспекти водійської освіти. *Журнал соціологічних спостережень*. Vol. 20. URL: <http://habitus.od.ua/journals/2020/20-2020/33.pdf> (дата звернення: 04.08.2025).
43. ESI Tu Solutions. Assess & train in virtual reality: Driver education applications. URL: <https://esitutionsolutions.com/assess-train-in-virtual-reality/> (дата звернення: 04.08.2025).
44. Transportation Research Part F: Traffic Psychology and Behaviour. (2025). Innovative approaches to driver training. URL: <https://www.sciencedirect.com/science/article/pii/S1369847825000579> (дата звернення: 04.08.2025).
45. ImmerSafety. Defensive VR driving simulator: Advanced training solutions. URL: <https://immersafety.com/defensive-vr-driving-simulator/> (дата звернення: 04.08.2025).
46. Goleman D. *Focus: The hidden driver of excellence*. Harper. 2013.
47. Green M. Driver reaction time. Visual Expert. URL: <https://www.visualexpert.com/Resources/reactiontime.html> (дата звернення: 04.08.2025).
48. International Security Driver Association. (2024). Reaction time in driving performance. URL: <https://isdacenter.org/reaction-time/> (дата звернення: 04.08.2025).
49. Drive in Control. (2024). What all drivers should know from the crash prevention experts. URL: <https://www.driveincontrol.org/tailgating> (дата звернення: 04.08.2025).
50. MDPI. Applied Sciences: Driver training technology innovations. URL: <https://www.mdpi.com/2032-6653/15/10/474> (дата звернення: 04.08.2025).

51. Sweller J. Cognitive load during problem solving: Effects on learning. *Cognitive Science*. 1988. Vol. 12, No. 2. P. 257–285. DOI: 10.1207/s15516709cog1202_4.

52. Miller G. A. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*. 1956. Vol. 63, No. 2. P. 81–97. DOI: 10.1037/h0043158.

53. Cowan N. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences*. 2001. Vol. 24, No. 1. P. 87–114. DOI: 10.1017/s0140525x01003922.

54. Cowan N. The magical mystery four: How is working memory capacity limited, and why does it matter? *Current Directions in Psychological Science*. 2010. Vol. 19, No. 1. P. 51–57.

55. Sweller J. Element interactivity and intrinsic, extraneous, and germane cognitive load. *Educational Psychology Review*. 2010. Vol. 22, No. 2. P. 123–138.

56. Sweller J., Ayres P., Kalyuga S. *Cognitive load theory*. Springer. 2011. DOI: 10.1007/978-1-4419-8126-4.

57. Transportation Research Interdisciplinary Perspectives. (2023). Modern approaches to driver education. URL: <https://www.sciencedirect.com/science/article/pii/S2590198223000520> (дата звернення: 04.08.2025).

58. Pixaera. Driving safety training: Modern technological approaches. URL: <https://www.pixaera.com/driving-safety-training/> (дата звернення: 04.08.2025).

59. von Beesten S., Bresges A. The effectiveness of teaching methods for preparing teacher education students to teach road safety-related content. *Frontiers in Education*. 2025. Vol. 9. DOI: 10.3389/feduc.2024.1471022.

60. Engström, J. (2018). Effects of cognitive load and anticipation on driver responses to a critical traffic event. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 62(1), 1584–1588. SAGE Publications. DOI: 10.1177/1541931218621358.

61. Growth Engineering. (2024, February 13). Cognitive load theory: The secret to managing mental overload. URL: <https://www.growthengineering.co.uk/cognitive-load-theory/> (дата звернення: 04.08.2025).

62. Abrahamse E. L., Verwey W. B. Context-dependent motor skill and the role of practice. *Psychonomic Bulletin & Review*. 2008. Vol. 15, No. 6. P. 1058–1064. URL: <https://pmc.ncbi.nlm.nih.gov/articles/PMC3470693/> (дата звернення: 04.08.2025).

63. Müssgens D. M., Ullén F. Transfer in motor sequence learning: Effects of practice schedule and sequence context. *Frontiers in Human Neuroscience*. 2015. Vol. 9. Article 642. URL: <https://pmc.ncbi.nlm.nih.gov/articles/PMC4656827/> (дата звернення: 04.08.2025).

64. Weiße B., et al. *Theory and practice in driver education*. Tallinn University. 2015. URL: https://www.tlu.ee/sites/default/files/Haapsalu_kolledž/Projektide_lisainfo/Theory_and_Practice_in_Driver_Education.pdf (дата звернення: 04.08.2025).

65. Salmoni A. W., et al. Cognitive workload and the driver. DiVA Portal. URL: <http://www.diva-portal.org/smash/get/diva2:196947/FULLTEXT01.pdf> (дата звернення: 04.08.2025).

66. Komersant Ukrainian. (2025). Problems with driver training: Only 18% pass the exam on the first attempt. URL: <https://www.komersant.info/en/problem-y-z-pidhotovko-iu-vodiiv-lyshe-18-skladaie-ispyt-z-pershoi-sproby/> (дата звернення: 04.08.2025).

67. Department for Transport. (2024). Driver and rider testing and instructor statistics: April 2023 to March 2024. GOV.UK. URL: <https://www.gov.uk/government/statistics/driver-and-rider-testing-and-instructor-statistics-april-2023-to-march-2024/driver-and-rider-testing-and-instructor-statistics-april-2023-to-march-2024> (дата звернення: 04.08.2025).

68. Garlan, D. (2000). Software architecture: a roadmap. In A. Finkelstein (Ed.), *The Future of Software Engineering: Proceedings of the Conference on The*

Future of Software Engineering (pp. 91–101). ACM Press.
<https://doi.org/10.1145/336512.336537>

69. Richards, M., & Ford, N. (2020). *Fundamentals of software architecture: An engineering approach*. O'Reilly Media. ISBN: 978-1-492-04344-7.
<https://www.oreilly.com/library/view/fundamentals-of-software/9781492043447/>

70. Newman, S. (2015). *Building microservices: Designing fine-grained systems*. O'Reilly Media.

71. Shemshack, A., & Spector, J. M. (2020). A systematic literature review of personalized learning terms. *Smart Learning Environments*, 7(1), 33.
<https://doi.org/10.1186/s40561-020-00140-9>

72. Chen, X., Zou, D., Xie, H., Cheng, G., & Liu, C. (2022). Two decades of artificial intelligence in education. *Educational Technology & Society*, 25(1), 28–47.
<https://www.jstor.org/stable/48647028>

73. Henning, P. A., Heberle, F., Streicher, A., Wolf, R., Swertz, C., Fanzott, B., Zander, S., & Korica-Pehserl, P. (2015). Personalized web learning: Merging open educational resources with personal learning environments. *International Journal of Emerging Technologies in Learning (iJET)*, 10(3), 14–22.
<https://doi.org/10.3991/ijet.v10i3.4752>

74. Advanced Distributed Learning (ADL). (2009). SCORM 2004 4th Edition overview. ADL.
https://www.adlnet.gov/assets/uploads/SCORM_2004_4ED_v1_1_TR_20090814.pdf

75. IMS Global Learning Consortium. (2019). Learning Tools Interoperability (LTI) core specification version 1.3. IMS Global.
<https://www.imsglobal.org/spec/lti/v1p3/>

76. Dougiamas, M., & Taylor, P. C. (2003). Moodle: Using learning communities to create an open source course management system. In *Proceedings of the EDMEDIA 2003 Conference* (pp. 171–178). Association for the Advancement of Computing in Education. <https://dougiamas.com/edmedia2003/>

77. Cavus, N. (2015). Distance learning and learning management systems. *Procedia – Social and Behavioral Sciences*, 191, 872–877. <https://doi.org/10.1016/j.sbspro.2015.04.611>
78. Horton, W. (2011). *E-learning by design* (2nd ed.). Pfeiffer/Wiley.
79. Graf, S., & List, B. (2005). An evaluation of open source e-learning platforms stressing adaptation issues. In *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies (ICALT 2005)* (pp. 163–165). IEEE. <https://doi.org/10.1109/ICALT.2005.54>
80. Benta, D., Bologa, G., & Dzitac, I. (2014). E-learning platforms in higher education. Case study. *Procedia Computer Science*, 31, 1170–1176. DOI: 10.1016/j.procs.2014.05.373.
81. Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices architecture enables DevOps: Migration to a cloud-native architecture. *IEEE Software*, 33(3), 42–52. <https://doi.org/10.1109/MS.2016.64>
82. Pahl, C., Brogi, A., Soldani, J., & Jamshidi, P. (2019). Cloud container technologies: A state-of-the-art review. *IEEE Transactions on Cloud Computing*, 7(3), 677–692. <https://doi.org/10.1109/TCC.2017.2702586>
83. Taibi, D., Lenarduzzi, V., & Pahl, C. (2018). Architectural patterns for microservices: A systematic mapping study. In *Proceedings of the 8th International Conference on Cloud Computing and Services Science (CLOSER 2018)* (pp. 221–232). SciTePress. <https://doi.org/10.5220/0006798302210232>
84. Dillenbourg, P., Schneider, D., & Synteta, P. (2002). Virtual learning environments. In *Proceedings of the 3rd Hellenic Conference on Information and Communication Technologies in Education* (pp. 3–18). Kastaniotis Editions.
85. Reich, J., & Ruipérez-Valiente, J. A. (2019). The MOOC pivot. *Science*, 363(6423), 130–131. <https://doi.org/10.1126/science.aav7958>
86. Klašnja-Milićević, A., Vesin, B., Ivanović, M., & Budimac, Z. (2011). E-learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Education*, 56(3), 885–899. <https://doi.org/10.1016/j.compedu.2010.11.001>

87. Tang, T. Y., & McCalla, G. (2005). Smart recommendation for an evolving e-learning system: Architecture and experiment. *International Journal on E-Learning*, 4(1), 105–129.
88. Thönes, J. (2015). Microservices. *IEEE Software*, 32(1), 116–116. <https://doi.org/10.1109/MS.2015.11>
89. Brusilovsky, P. (2001). Adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 11(1–2), 87–110. <https://doi.org/10.1023/A:1011143116306>
90. Brusilovsky, P., & Millán, E. (2007). User models for adaptive hypermedia and adaptive educational systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl (Eds.), *The Adaptive Web* (Lecture Notes in Computer Science, Vol. 4321, pp. 3–53). Springer. https://doi.org/10.1007/978-3-540-72079-9_1
91. Gašević, D., Djurić, D., & Devedžić, V. (2006). *Model driven engineering and ontology development* (2nd ed.). Springer.
92. Chen, C.-M., & Duh, L.-J. (2008). Personalized web-based tutoring system based on fuzzy item response theory. *Expert Systems with Applications*, 34(4), 2298–2315. <https://doi.org/10.1016/j.eswa.2007.03.010>
93. Kobsa, A. (2007). Generic user modeling systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl (Eds.), *The Adaptive Web* (Lecture Notes in Computer Science, Vol. 4321, pp. 136–154). Springer. https://doi.org/10.1007/978-3-540-72079-9_4
94. Brusilovsky, P., & Peylo, C. (2003). Adaptive and intelligent web-based educational systems. *International Journal of Artificial Intelligence in Education*, 13(2–4), 159–172.
95. Kulik, J. A., & Fletcher, J. D. (2016). Effectiveness of intelligent tutoring systems: A meta-analytic review. *Review of Educational Research*, 86(1), 42–78. DOI: 10.3102/0034654315581420.
96. Nwana, H. S. (1990). Intelligent tutoring systems: An overview. *Artificial Intelligence Review*, 4(4), 251–277. <https://doi.org/10.1007/BF00168958>
97. Dagger, D., O'Connor, A., Lawless, S., Walsh, E., & Wade, V. P. (2007). Service-oriented e-learning platforms: From monolithic systems to flexible

services. *IEEE Internet Computing*, 11(3), 28–35.
<https://doi.org/10.1109/MIC.2007.70>

98. Shoham, Y., & Leyton-Brown, K. (2009). *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press.
<https://doi.org/10.1017/CBO9780511811654>

99. Manouselis, N., Drachsler, H., Vuorikari, R., Hummel, H. G. K., & Koper, R. (2011). Recommender systems in technology enhanced learning. In P. B. Kantor, F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender Systems Handbook* (pp. 387–415). Springer. https://doi.org/10.1007/978-0-387-85820-3_12

100. Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative filtering recommender systems. In P. Brusilovsky, A. Kobsa, & W. Nejdl (Eds.), *The Adaptive Web* (Lecture Notes in Computer Science, Vol. 4321, pp. 291–324). Springer. https://doi.org/10.1007/978-3-540-72079-9_9

101. Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H. H., Ventura, M., Olney, A., & Louwerse, M. M. (2004). AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2), 180–193.
<https://doi.org/10.3758/BF03195563>

102. VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4), 197–221. <https://doi.org/10.1080/00461520.2011.611369>

103. Ritter, S., Anderson, J. R., Koedinger, K. R., & Corbett, A. (2007). Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*, 14(2), 249–255. <https://doi.org/10.3758/BF03194060>

104. Drachsler, H., Verbert, K., Santos, O. C., & Manouselis, N. (2015). Panorama of recommender systems to support learning. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender Systems Handbook* (2nd ed., pp. 421–451). Springer.
https://doi.org/10.1007/978-1-4899-7637-6_12

105. Baker, R. S. J. d., & Yacef, K. (2009). The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining*, 1(1), 3–17.
<https://doi.org/10.5281/zenodo.3554657>

106. Lam, X. N., Vu, T., Le, T. D., & Duong, A. D. (2008). Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication (ICUIMC '08)* (pp. 208–211). ACM. <https://doi.org/10.1145/1352793.1352837>
107. Conati, C., Porayska-Pomsta, K., & Mavrikis, M. (2018). AI in education needs interpretable machine learning: Lessons from open learner modelling. *arXiv preprint*. <https://arxiv.org/abs/1807.00154>
108. Woolf, B. P. (2009). *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann / Elsevier. <https://doi.org/10.1016/B978-0-12-373594-2.X0001-1>
109. Siemens, G., & Baker, R. S. J. d. (2012). Learning analytics and educational data mining: Towards communication and collaboration. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge (LAK '12)* (pp. 252–254). ACM. <https://doi.org/10.1145/2330601.2330661>
110. Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4), 253–278. <https://doi.org/10.1007/BF01099821>
111. Piech, C., Spencer, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. *Advances in Neural Information Processing Systems*, 28, 505–513. <https://proceedings.neurips.cc/paper/2015/hash/bac9162b47c56fc8a4d2a519803d51b3-Abstract.html>
112. VanLehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16(3), 227–265. DOI: 10.3233/IRG-2006-16(3)02.
113. Chandler, P., & Sweller, J. (1992). The split-attention effect as a factor in the design of instruction. *British Journal of Educational Psychology*, 62(2), 233–246. DOI: 10.1111/j.2044-8279.1992.tb01017.x.

114. Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, 63(2), 81–97. <https://doi.org/10.1037/h0043158>
115. Cowan, N. (2001). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and Brain Sciences*, 24(1), 87–114. <https://doi.org/10.1017/S0140525X01003922>
116. Sweller, J., van Merriënboer, J. J. G., & Paas, F. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, 10(3), 251–296. <https://doi.org/10.1023/A:1022193728205>
117. Schneider, W., & Shiffrin, R. M. (1977). Controlled and automatic human information processing: I. Detection, search, and attention. *Psychological Review*, 84(1), 1–66. <https://doi.org/10.1037/0033-295X.84.1.1>
118. Csikszentmihalyi, M. (1990). *Flow: The psychology of optimal experience*. Harper & Row.
119. Paas, F., Tuovinen, J. E., Tabbers, H., & van Gerven, P. W. M. (2003). Cognitive load measurement as a means to advance cognitive load theory. *Educational Psychologist*, 38(1), 63–71. https://doi.org/10.1207/S15326985EP3801_8
120. Antonenko, P., Paas, F., Grabner, R., & van Gog, T. (2010). Using electroencephalography to measure cognitive load. *Educational Psychology Review*, 22(4), 425–438. <https://doi.org/10.1007/s10648-010-9130-y>
121. Arns, M., & Breteler, M. (2022). EEG-based prediction of cognitive load: Single-trial prediction with high accuracy. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 30, 1487–1496. <https://doi.org/10.1109/TNSRE.2022.3173624>
122. Mulder, L. J. M. (1992). Measurement and analysis methods of heart rate and respiration for use in applied environments. *Biological Psychology*, 34(2–3), 205–236. [https://doi.org/10.1016/0301-0511\(92\)90016-N](https://doi.org/10.1016/0301-0511(92)90016-N)
123. Beatty, J. (1982). Task-evoked pupillary responses, processing load, and the structure of processing resources. *Psychological Bulletin*, 91(2), 276–292. <https://doi.org/10.1037/0033-2909.91.2.276>

124. Herff, C., Heger, D., Fortmann, O., Hennrich, J., Putze, F., & Schultz, T. (2014). Mental workload during n-back task quantified in the prefrontal cortex using fNIRS. *Frontiers in Human Neuroscience*, 7, 935. <https://doi.org/10.3389/fnhum.2013.00935>
125. Paquette, L., & Baker, R. S. (2019). Comparing machine learning to knowledge engineering for student behavior modeling. *International Journal of Artificial Intelligence in Education*, 29(2), 281–304. <https://doi.org/10.1007/s40593-018-0170-8>
126. Hayhoe, M., & Ballard, D. (2005). Eye movements in natural behavior. *Trends in Cognitive Sciences*, 9(4), 188–194. <https://doi.org/10.1016/j.tics.2005.02.009>
127. Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. A. Hancock & N. Meshkati (Eds.), *Human Mental Workload* (pp. 139–183). North-Holland Press. <https://humansystems.arc.nasa.gov/groups/TLX/publications.php>
128. Paas, F. G. W. C. (1992). Training strategies for attaining transfer of problem-solving skill in statistics: A cognitive-load approach. *Journal of Educational Psychology*, 84(4), 429–434. <https://doi.org/10.1037/0022-0663.84.4.429>
129. Strayer, D. L., & Drews, F. A. (2007). Cell-phone-induced driver distraction. *Current Directions in Psychological Science*, 16(3), 128–131. <https://doi.org/10.1111/j.1467-8721.2007.00489.x>
130. Lemaignan, S., Jacq, A., Hood, D., Garcia, F., Paiva, A., & Dillenbourg, P. (2016). Learning by teaching a robot: The case of handwriting. *IEEE Robotics & Automation Magazine*, 23(2), 56–66. <https://doi.org/10.1109/MRA.2016.2546700>
131. Fleming, N. D., & Mills, C. (1992). Not another inventory, rather a catalyst for reflection. *To Improve the Academy*, 11(1), 137–155. <https://doi.org/10.1002/j.2334-4822.1992.tb00213.x>
132. Csikszentmihalyi, M. (1997). *Finding flow: The psychology of engagement with everyday life*. Basic Books.

133. Mavrikis, M., Rummel, N., & McLaren, B. M. (2021). Combining knowledge tracing and cognitive load for adaptive learning. *Computers in Human Behavior*, 124, 106891. <https://doi.org/10.1016/j.chb.2021.106891>
134. Bauer, M., & Nückles, M. (2021). Adaptive microlearning for reducing cognitive overload: A field study. *Computers & Education*, 173, 104284. <https://doi.org/10.1016/j.compedu.2021.104284>
135. Sriramprakash, S., Prasanna, V. D., & Murthy, O. V. R. (2017). Stress detection in working people. *Procedia Computer Science*, 115, 359–366. <https://doi.org/10.1016/j.procs.2017.09.090>
136. Matthews, G., Sparkes, T. J., & Bygrave, H. M. (1996). Attentional overload, stress, and simulated driving performance. *Human Performance*, 9(1), 77–101. https://doi.org/10.1207/s15327043hup0901_4
137. Bella, F., & Russo, M. (2011). A driving simulator study: Herzberg's two-factor theory applied to the driving task. *Transportation Research Part F: Traffic Psychology and Behaviour*, 14(1), 1–12. <https://doi.org/10.1016/j.trf.2010.09.003>
138. Young, M. S., & Stanton, N. A. (2002). Malleable attentional resources theory: A new explanation for the effects of mental underload on performance. *Human Factors*, 44(3), 365–375. <https://doi.org/10.1518/0018720024497709>
139. Katuhara, M., Shimizu, Y., & Marques, S. (2019). FORUM8 UC-win/Road: An Integrated 3D Virtual Environment for Urban Infrastructure and Driver Training Simulation. *Urban Computing and Visualization (Urbancomp)*, 26th International Conference. IEEE Xplore.
140. Lopez, P. A., Wiessner, E., Behrisch, M., Bieker-Walz, L., Erdmann, J., et al. (2018). Microscopic Traffic Simulation using SUMO. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)* (pp. 2575–2582). IEEE. DOI: 10.1109/ITSC.2018.8569938.
141. Nassri, H., & Hondori, H. (2018). A Systematic Review on Adaptive Virtual Reality for Training and Education. In *International Conference on Augmented Reality, Virtual Reality and Computer Graphics* (pp. 231–246). Springer, Cham. DOI: 10.1007/978-3-030-01054-6_14.

142. Schneegass, S., Pfleging, B., Broy, N., Heinrich, F., & Schmidt, A. (2013). A data set of real world driving to assess driver workload. In *Proceedings of the 5th International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '13)* (pp. 150–157). ACM. DOI: 10.1145/2516540.2516561.
143. Arutyunova, K. R., Bakhchina, A. V., Konovalov, D. I., Margaryan, M., Filimonov, A. V., & Shishalov, I. S. (2024). Heart rate dynamics for cognitive load estimation in a driving simulation task. *Scientific Reports*, 14, 31656. DOI: 10.1038/s41598-024-79728-x.
144. Anderson, J. R. (2007). *How Can the Human Mind Occur in the Physical Universe?* Oxford University Press.
145. Hick, W. E. (1952). On the rate of gain of information. *The Quarterly Journal of Experimental Psychology*, 4(1), 11–26. DOI: 10.1080/17470215208416600.
146. Baddeley, A. (2003). Working memory: Looking back and looking forward. *Review of General Psychology*, 7(2), 85–100. DOI: 10.1037/1089-2680.7.2.85.
147. SAE International. (2022). *Architecture analysis & design language (AADL)* (Standard No. AS5506D). <https://www.sae.org/standards/content/AS5506D/>
148. Embretson, S. E., & Reise, S. P. (2000). *Item response theory for psychologists*. Psychology Press. <https://doi.org/10.4324/9781410605269>
149. Weiss, D. J., & Kingsbury, G. G. (1984). Application of computerized adaptive testing to educational problems. *Journal of Educational Measurement*, 21(4), 361–375. <https://doi.org/10.1111/j.1745-3984.1984.tb01040.x>
150. Boboc, R. G., Butilă, E. V., & Butnariu, S. (2024). Leveraging wearable sensors in virtual reality driving simulators: A review of techniques and applications. *Sensors*, 24(13), 4417. <https://doi.org/10.3390/s24134417>
151. Nasri, M. (2025). Towards intelligent VR training: A physiological adaptation framework for cognitive load and stress detection. *arXiv preprint arXiv:2504.06461*. <https://doi.org/10.48550/arXiv.2504.06461>

152. ДСТУ ISO/IEC/IEEE 42010:2018. (2018). *Інженерія систем і програмних засобів. Опис архітектури* (ISO/IEC/IEEE 42010:2011, IDT). Київ: Мінекономрозвитку України. https://online.budstandart.com/ua/catalog/doc-page.html?id_doc=77960

153. Rasch, G. (1960). *Probabilistic models for some intelligence and attainment tests*. Danish Institute for Educational Research.

154. Lord, F. M. (1980). *Applications of item response theory to practical testing problems*. Lawrence Erlbaum Associates.

155. Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinee's ability. In F. M. Lord & M. R. Novick (Eds.), *Statistical theories of mental test scores* (pp. 397–479). Addison-Wesley.

156. Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The Knowledge-Learning-Instruction (KLI) framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, 36(5), 757–798. <https://doi.org/10.1111/j.1551-6709.2012.01245.x>

157. Weiss, D. J. (1982). Improving measurement quality and efficiency with adaptive testing. *Applied Psychological Measurement*, 6(4), 473–492. <https://doi.org/10.1177/014662168200600408>

158. van der Linden, W. J., & Glas, C. A. W. (Eds.). (2010). *Elements of adaptive testing*. Springer. <https://doi.org/10.1007/978-0-387-85461-8>

159. Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2), 89–100. <https://doi.org/10.1111/j.1469-7610.1976.tb00381.x>

160. González-Calatayud, V., Prendes-Espinosa, P., & Roig-Vila, R. (2021). Artificial intelligence for student assessment: A systematic review. *Applied Sciences*, 11(12), 5467. <https://doi.org/10.3390/app11125467>

161. Boboc, R. G., Băutu, E., Gîrbacia, F., Popovici, N., & Popovici, D.-M. (2024). Adaptive learning in VR training: Balancing educational goals and cybersickness mitigation. *Electronics*, 13(2), 376. <https://doi.org/10.3390/electronics13020376>

162. Nasri, N., Orts-Escolano, S., Cazorla, M., & García-Rodríguez, J. (2025). Enhancing VR training through EEG-based cognitive load assessment and adaptive difficulty adjustment. *Computers in Human Behavior*, 162, 108438. <https://doi.org/10.1016/j.chb.2024.108438>
163. Sharma, K., Papamitsiou, Z., & Giannakos, M. (2020). Building pipelines for educational data using AI and multimodal analytics: A “grey-box” approach. *British Journal of Educational Technology*, 51(6), 2004–2031. <https://doi.org/10.1111/bjet.13039>
164. Shaffer, F., & Ginsberg, J. P. (2017). An overview of heart rate variability metrics and norms. *Frontiers in Public Health*, 5, 258. <https://doi.org/10.3389/fpubh.2017.00258>
165. Malik, M., Bigger, J. T., Camm, A. J., Kleiger, R. E., Malliani, A., Moss, A. J., & Schwartz, P. J. (1996). Heart rate variability: Standards of measurement, physiological interpretation, and clinical use. *Circulation*, 93(5), 1043–1065. <https://doi.org/10.1161/01.CIR.93.5.1043>
166. Kim, H. G., Cheon, E. J., Bai, D. S., Lee, Y. H., & Koo, B. H. (2018). Stress and heart rate variability: A meta-analysis and review of the literature. *Psychiatry Investigation*, 15(3), 235–245. <https://doi.org/10.30773/pi.2017.08.17>
167. Aljunid, M. F., Manjaiah, D. H., Hooshmand, M. K., Ali, W. A., Shetty, A. M., & Alzoubah, S. Q. (2025). A collaborative filtering recommender systems: Survey. *Neurocomputing*, 617, 128718. <https://doi.org/10.1016/j.neucom.2024.128718>
168. Cimatti, A., Clarke, E. M., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., & Tacchella, A. (2002). NuSMV 2: An open-source tool for symbolic model checking. In E. Brinksma & K. G. Larsen (Eds.), *Computer Aided Verification (CAV 2002)* (Lecture Notes in Computer Science, Vol. 2404, pp. 359–364). Springer. https://doi.org/10.1007/3-540-45657-0_29
169. van der Aalst, W. M. P. (2016). *Process mining: Data science in action* (2nd ed.). Springer. <https://doi.org/10.1007/978-3-662-49851-4>

170. Leemans, S. J. J., Fahland, D., & van der Aalst, W. M. P. (2013). Discovering block-structured process models from event logs containing infrequent behaviour. In N. R. Brisaboa, O. Pichler, & A. Gilberto (Eds.), *Business Process Management Workshops* (pp. 66–78). Springer. https://doi.org/10.1007/978-3-319-06257-0_6

171. van der Aalst, W. M. P., Weijters, T., & Maruster, L. (2004). Workflow mining: Discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9), 1128–1142. <https://doi.org/10.1109/TKDE.2004.47>

172. Weijters, A. J. M. M., van der Aalst, W. M. P., & Alves de Medeiros, A. K. (2006). Process mining with the HeuristicsMiner algorithm (Technical Report WP 166). Eindhoven University of Technology.

173. Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J.-M., & Irwin, J. (1997). Aspect-oriented programming. In M. Aksit & S. Matsuoka (Eds.), *ECOOP'97 – Object-Oriented Programming* (Lecture Notes in Computer Science, Vol. 1241, pp. 220–242). Springer. <https://doi.org/10.1007/BFb0053381>

174. Berti, A., & van der Aalst, W. M. P. (2020). A novel token-based replay technique to speed up conformance checking and process enhancement. In M. Koutny, F. Kordon, & L. Pomello (Eds.), *Transactions on Petri Nets and Other Models of Concurrency XV* (pp. 1–26). Springer. https://doi.org/10.1007/978-3-662-63079-2_1

175. Sounderajah V. et al. Simulator Pre-Screening of Underprepared Drivers Prior to Licensing On-Road Examination: Clustering of Virtual Driving Test Time Series Data // Journal of Medical Internet Research. 2020. Vol. 22, No. 6. e13995. Доступ: <https://www.jmir.org/2020/6/e13995/>

176. Schmidt P., Reiss A., Duerichen R., Marberger C., Van Laerhoven K. Introducing WESAD, a Multimodal Dataset for Wearable Stress and Affect Detection // Proceedings of the 20th ACM International Conference on Multimodal Interaction (ICMI '18). 2018. P. 400–408. DOI: 10.1145/3242969.3242985. Доступ: <https://dl.acm.org/doi/10.1145/3242969.3242985>.

ДОДАТОК А

ПЕРЕЛІК ПУБЛІКАЦІЙ ЗДОБУВАЧА

1. Малюга А. І. Концептуальні засади дослідження впровадження технологій віртуальної реальності у навчальний процес автошкіл // *Вчені записки ТНУ імені В. І. Вернадського. Серія: Технічні науки*. – 2024. – Т. 35 (74), № 1. – С. 214–221. – DOI: 10.32782/2663-5941/2024.1.1/33.

2. Малюга А. І. Гібридна програмна архітектура з периферійними обчисленнями для адаптивних VR-систем навчання водінню з біометричним зворотним зв'язком // *Авіаційно-космічна техніка і технологія*. – 2025. – № 5. – С. 96–111. – DOI: 10.32620/aktt.2025.5.09.

3. Малюга А. І. Методи побудови адаптивного програмного забезпечення навчальних систем // *Авіаційно-космічна техніка і технологія*. – 2025. – № 6. – С. 113–127. – DOI: 10.32620/aktt.2025.6.11.

4. Малюга А. І. Метод онлайн-диспетчеризації тематичних модулів в адаптивних навчальних системах із використанням векторів індивідуального профілю та поточного стану користувача // *Авіаційно-космічна техніка і технологія*. – 2026. – № 1. – С. 123–132. – DOI: 10.32620/aktt.2026.1.11.

5. Малюга А. І., Соколова Є. В. Метод верифікації адаптивних програмних систем з використанням технології process mining // *Відкриті інформаційні та комп'ютерні інтегровані технології*. – 2026. – № 107. – С. 257–263. – DOI: 10.32620/oikit.2026.107.17.

6. Малюга А. І. Шляхи підвищення ефективності процесу навчання навичкам водіння автомобільної та спеціальної техніки за рахунок впровадження спеціалізованих програмних засобів // *Проблеми інформатизації: тези доповідей дванадцятої міжнародної науково-технічної конференції (21–22 листопада 2024 р.)*. – Баку – Харків – Бельсько-Бяла, 2024. – Т. 2, секція 4. – С. 28.

7. Малюга А. І. Особливості застосування штучних нейронних мереж у тренажерах для підготовки водіїв автомобільної та спеціальної техніки // *Сучасні*

напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей п'ятнадцятої міжнародної науково-технічної конференції (24–25 квітня 2025 р.). – Баку – Харків – Жиліна, 2025. – Т. 1, секції 1, 5. – С. 28.

8. Малюга А. І. Моделювання процесу обробки біометричних даних у середовищі віртуальної реальності при адаптивному навчанні водіїв // *Проблеми інформатизації: тези доповідей тринадцятої міжнародної науково-технічної конференції (27–28 листопада 2025 р.).* – Баку – Харків – Бельсько-Бяла, 2025. – Т. 1, секції 1, 2. – С. 36.

9. Малюга А. І. Верифікація адаптивних програмних систем, що змінюють поведінку у відповідності до поточного психофізіологічного стану користувача // *Сучасні напрями розвитку інформаційно-комунікаційних технологій та засобів управління: тези доповідей шістнадцятої міжнародної науково-технічної конференції (23–24 квітня 2026 р.).* – Баку – Харків – Жиліна, 2026. – Т. 1, секція 2. – С. 43.

10. Шостак І. В., Малюга А. І. Методичні і програмні засоби персоніфікованої адаптації навчального процесу в автошколі // *Інтегровані інтелектуальні робототехнічні комплекси (ІРТК-2026): матеріали дев'ятнадцятої міжнародної науково-практичної конференції (19–20 травня 2026 р.).* – Київ: НАУ, 2026. – С. 557-558.

ДОДАТОК Б

ДОВІДКА ПРО ВПРОВАДЖЕННЯ РЕЗУЛЬТАТІВ ДИСЕРТАЦІЙНОГО ДОСЛІДЖЕННЯ

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ

«Дев-Оптіма»

Україна, 61100, Харківська обл., м. Харків,

вул. Каденюка, буд. 16, кв. 15.

Ідентифікаційний код юридичної особи 45747392,

Код ЄДР 1010351020000015158

Види економічної діяльності:

62.01 Комп'ютерне програмування (основний);

63.99 Надання інших інформаційних послуг;

63.12 Веб-портали;

62.09 Інша діяльність у сфері інформаційних технологій і комп'ютерних систем.



ДОВІДКА

про використання наукових і прикладних результатів дисертаційного дослідження
Малюги Артура Івановича «Модель та методи персоніфікованої адаптації процесу
підготовки водіїв в навчальних програмних системах автошкіл» у діяльності
ТОВ «Дев-Оптіма»

Підтверджуємо, що у період з 5 по 8 січня 2026 року нами було одержано від Малюги А.І., аспіранта кафедри інженерії програмного забезпечення Національного аерокосмічного університету «Харківський авіаційний інститут» такі матеріали, що у сукупності являють собою методичний базис розроблення типової програмної навчальної системи для інформаційного підтримування процесу підготовки водіїв у спеціалізованих навчальних закладах:

- модель гібридної програмної архітектури з периферійними обчисленнями системи навчання в автошколі, яка містить клієнтський рівень для візуалізації дорожнього середовища, периферійний рівень для диспетчеризації персоніфікованого вибору навчальних завдань та мікросервісний рівень для верифікації вибору навчальних завдань, що створює умови для персоніфікованої адаптації процесу навчання в автошколі;

- метод персоніфікованої адаптації процесу навчання, шляхом вибору навчальних модулів на основі векторів профілю і стану користувача з урахуванням ступеню новизни завдань та попередньої фільтрації модулів за рівнем складності, що дає можливість у автоматизованому режимі підібрати навчальний сценарій згідно рівня підготовки та поточного когнітивного навантаження користувача навчальної програмної системи;

- метод верифікації процесу підготовки водіїв в навчальних програмних системах автошкіл, який містить етапи побудови моделей цільового та фактичного процесів персоніфікованої адаптації навчання та виявлення порушень адаптивної логіки на основі порівняння цільової та фактичної моделей, що забезпечує умови для удосконалення правил персоніфікованої адаптації у процесі навчання з урахуванням рівня латентних знань користувачів.

Зазначені матеріали використовуються у ході реалізації проєкту створення типового комплексу програмних засобів інформаційного підтримування процесу підготовки водіїв у автошколах, на засадах персоніфікованої адаптації.

Директор ТОВ «Дев-Оптіма»

Юрій АБЕРІН

28 квітня 2026 р.