УДК 004.9

ІНТЕЛЕКТУАЛЬНІ РЕКОМЕНДАЦІЙНІ СИСТЕМИ: ПОЄДНАННЯ ДІАГНОСТИКИ НЕСПРАВНОСТЕЙ ПЕРСОНАЛЬНИХ КОМП'ЮТЕРІВ ТА АНАЛІЗУ КОРИСТУВАЦЬКОЇ ВЗАЄМОДІЇ

І. В. Оврамець, Б. А. Явгусішин, Ю. С. Антонов

Анотація. У статті розглянуто підхід до побудови інтелектуальної рекомендаційної системи, що поєднує діагностику несправностей персональних комп'ютерів із аналізом користувацької взаємодії. Для технічної діагностики використано C#, MAUI, ML.NET та EF Core, а для аналітичної підсистеми – Python, Django і PostgreSQL. Запропонована архітектура дозволяє об'єднати результати технічного моніторингу та поведінкового аналізу для формування персоналізованих рекомендацій. Очікується, що такий підхід сприятиме підвищенню точності прогнозування несправностей і релевантності рекомендацій порівняно з традиційними рішеннями.

Ключові слова: рекомендаційна система, машинне навчання, діагностика, користувацька взаємодія, мікросервіси.

Вступ. Сучасні персональні комп'ютери є складними системами, поєднання апаратного та програмного забезпечення яких вимагає ефективних методів діагностики та підтримки працездатності. У традиційних підходах процес виявлення несправностей базується на технічних параметрах і часто не враховує поведінку користувача, що знижує точність рекомендацій і ускладнює адаптацію рішень до конкретних умов експлуатації. Останніми роками у наукових публікаціях спостерігається зростання інтересу до використання штучного інтелекту й машинного навчання в задачах технічної діагностики та підтримки користувачів. Зокрема, розробки на основі ML.NET, TensorFlow і Scikit-learn демонструють ефективність у прогнозуванні збоїв і класифікації технічних станів. Водночас активно розвиваються рекомендаційні системи, що аналізують поведінкові дані користувачів для підвищення рівня персоналізації. Попри наявність окремих рішень у цих напрямах, питання інтеграції технічної діагностики з поведінковим аналізом залишаються недостатньо дослідженими. Відсутність єдиного архітектурного рішення, здатного поєднати обидва підходи, обмежує можливості побудови універсальних інтелектуальних систем підтримки користувачів.

Аналіз останніх досліджень і публікацій. Рекомендаційні системи та системи підтримки ухвалення рішень є важливою складовою багатьох програмних рішень. Завдяки цьому, у сучасних дослідженнях спостерігається активний розвиток методів інтелектуальної діагностики та оптимізації технічних процесів. У роботах С. Д. Штовби, О. Д. Панкевича та О. П. Ротштейна запропоновано багатокритеріальні підходи до побудови нечітких баз знань і застосування евристичних та генетичних алгоритмів у багатовимірному просторі типів дефектів [1, 2]. Такі рішення поєднують класичні моделі з інтелектуальними методами, забезпечуючи точність, компактність і надійність систем підтримки ухвалення рішень. У роботі [3] запропоновано експертну систему, яка дозволяє визначати частково правильні або вгадані відповіді та ухвалювати рішення про необхідність відкидання таких відповідей та формування нових питань. Система базується на використанні декартової відстані та відстані Лєвенштейна. У роботі М. Коллінза розглянуто застосування методів аналізу даних для виявлення аномалій та підвищення безпеки комп'ютерних систем [4]. Автор підкреслює важливість поведінкового аналізу та побудови моделей, здатних виявляти відхилення у роботі мережі. Запропоновані підходи демонструють ефективність використання аналітики даних для прогнозування збоїв і можуть бути адаптовані до задач технічної діагностики.

Також серед сучасних досліджень варто звернути увагу на роботу Д. Яннаха «A survey on multi-objective recommender systems», у якій узагальнено методи оптимізації рекомендаційних систем за кількома критеріями: точністю, різноманітністю та задоволеністю користувача [5]. У праці М. Фернандеса та співавторів «Machine learning techniques applied to mechanical fault diagnosis and fault prognosis in the context of real industrial manufacturing use-cases: a systematic literature review» розглянуто використання машинного навчання для діагностики технічних несправностей і прогнозування стану обладнання [6]. Дослідження К. Захаріадеса «A Review

of Artificial Intelligence Techniques in Fault Diagnosis of Electric Machines» узагальнює інтелектуальні методи діагностики електричних машин, акцентуючи увагу на перевагах гібридних моделей і нейронних підходів [7].

Метою статті є розроблення концепції інтегрованої інтелектуальної рекомендаційної системи, що поєднує модуль діагностики несправностей персональних комп'ютерів, реалізований із використанням C#, MAUI, ML.NET, EF Core, та окремий модуль аналізу користувацької взаємодії із MAUI застосунком на базі Python, Django, PostgreSQL для підвищення точності прогнозування несправностей і релевантності рекомендацій.

Виклад основного матеріалу. У межах описаної концепції у нашій статті розроблено архітектуру, що поєднує дві взаємопов'язані, але повністю автономні підсистеми (рис. 1). Основну частину системи будемо реалізовувати у вигляді кросплатформного застосунку на C# MAUI, який забезпечує виконання процесів технічної діагностики та містить структуровану базу даних із наперед визначеними сценаріями виявлення, аналізу й усунення несправностей [8]. Ця база даних формується на основі напрацьованих знань про типові технічні проблеми, алгоритми їх розпізнавання та перевірені способи усунення. Для реалізації логіки зберігання та доступу до даних використано EF Core, що забезпечує ефективну ORM-взаємодію з базою. Моделі машинного навчання, що створені за допомогою ML.NET, застосовуються для автоматичного визначення категорії несправності, аналізу ймовірних причин і формування базового переліку рішень.

Окремий модуль рекомендаційної системи реалізований із використанням Python, Django та PostgreSQL [9]. Його ключовою особливістю є можливість роботи в ізольованому середовищі Docker ,що гарантує стабільність, масштабованість і простоту розгортання [10, 11]. Цей сервіс не дублює дані MAUI-застосунку, а натомість звертається до вже існуючої бази знань, використовуючи стандартизований REST API. Такий підхід дозволяє мінімізувати навантаження на основну систему, оскільки рекомендаційний модуль не виконує повноцінного аналізу несправностей, а здійснює вибір оптимальних рішень із наявної бази на основі даних, отриманих від MAUI-застосунку. Для взаємодії цих двох підсистем між собою передбачається використання REST API за допомогою бібліотеки Refit, що спрощує інтеграцію, автоматично генерує HTTP-запити на основі інтерфейсів та мінімізує ризик помилок під час викликів API.

Отже, процес взаємодії відбувається за принципом взаємодоповнення. Після виконання діагностики MAUI-застосунок надсилає структуровані результати до рекомендаційного сервісу. Останній, маючи доступ до бази знань, що зберігаються у MAUI-додатку, обирає найбільш релевантні варіанти рекомендацій і повертає їх у зручному для відображення форматі. За рахунок цього частина розрахунків проводиться на окремій системі, що у свою чергу економить ресурси користувацького пристрою.

Архітектурне рішення, засноване на контейнеризації, модульності та спільному використанні бази знань, забезпечує гнучкість, розширюваність і можливість подальшої інтеграції з іншими сервісами. Розділення обов'язків між MAUI-застосунком і додатковою рекомендаційною підсистемою створює баланс між локальною аналітикою та віддаленою обробкою запитів, що дозволяє оптимізувати роботу системи без втрати продуктивності.

REST API забезпечує чітке розмежування функціональних рівнів, що спрощує підтримку та модернізацію системи. Кожен ендпойнт відповідає певній логічній операції – отриманню результатів діагностики, пошуку рекомендацій, формуванню аналітичного звіту тощо. Такий підхід робить систему більш прозорою, з точки зору логіки обробки запитів, та дозволяє легко відстежувати її роботу. До того ж REST API сприяє підвищенню безпеки взаємодії, оскільки передбачає використання стандартних механізмів аутентифікації на рівні взаємодії з модулем та шифрування запитів, що є важливим аспектом у роботі з технічними та користувацькими даними (рис. 1).

Важливим елементом у контексті взаємодії MAUI-застосунку з рекомендаційним модулем є використання бібліотеки Refit, яка суттєво спрощує роботу з REST API у середовищі .NET. Refit реалізує концепцію безпечних інтерфейсів, де структура HTTP-запитів описується у вигляді C# інтерфейсів із відповідними атрибутами. Це дозволяє автоматично генерувати

клієнтський код, зменшуючи кількість ручних операцій і потенційних помилок під час розробки. Розробнику достатньо описати методи з відповідними параметрами та атрибутами запиту, після чого Refit самостійно створює необхідну логіку взаємодії з REST API [12].

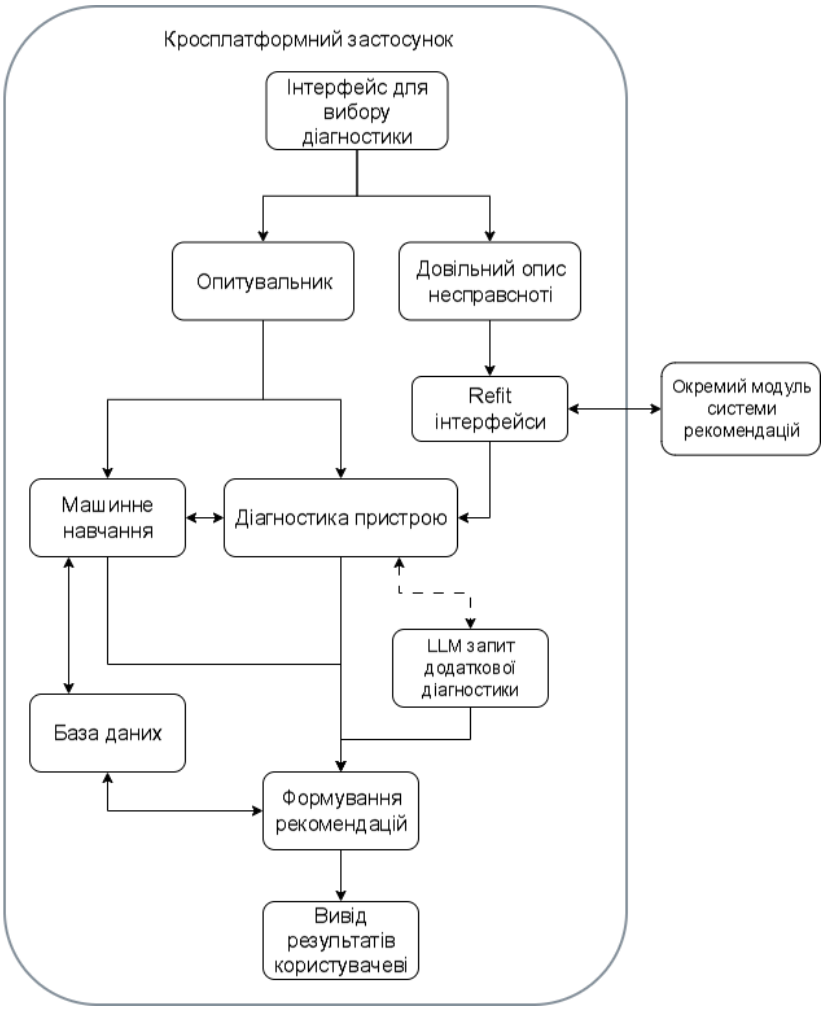

Рис. 1. Взаємодія кросплатформного застосунку із модуля аналізу користувацької взаємодії

Використання Refit суттєво підвищує читабельність коду та його підтримуваність. Будь-які зміни в API відображаються безпосередньо у визначеннях інтерфейсів, що полегшує оновлення клієнтської частини у разі модифікації серверної логіки. До того ж бібліотека забезпечує асинхронну взаємодію, що дозволяє виконувати HTTP-запити без блокування основного потоку, підвищуючи продуктивність і швидкість відгуку застосунку.

У поєднанні з Docker-контейнеризацією така архітектура забезпечує стабільну та передбачувану роботу системи в різних середовищах. REST API виступає універсальним інтерфейсом зв'язку, а Refit – інструментом, що робить взаємодію максимально простою, безпечною та ефективною (рис. 2). Це дозволяє зменшити кількість ручної інтеграційної логіки, знизити ризики виникнення помилок під час обміну даними та забезпечити легке масштабування – як шляхом додавання нових функціональних ендпоїнтів, так і через розширення системи іншими мікросервісами.

У перспективі така архітектура може бути розширена шляхом підключення нових джерел даних або аналітичних модулів для побудови повноцінного комплексу підтримки технічних рішень. Поєднання бази знань із гнучким рекомендаційним модулем на основі REST API створює основу для побудови сучасних адаптивних систем, здатних навчатися на реальних сценаріях використання та постійно підвищувати точність своїх рекомендацій.

Рис. 2. Взаємодія модуля аналізу користувацької взаємодії із кросплатформним застосунком

Висновки. Розроблена концепція поєднує технічну діагностику несправностей персональних комп'ютерів із рекомендаційними механізмами, що базуються на аналізі користувацької взаємодії. Запропонована архітектура передбачає взаємодію між кросплатформним застосунком розробленому на .NET MAUI та аналітичним сервісом, реалізованими у середовищі Django, через REST API з використанням бібліотеки Refit. Такий підхід забезпечує модульність, гнучкість і можливість для подальшого масштабування системи. Очікується, що реалізація запропонованої архітектури сприятиме підвищенню точності прогнозування несправностей і релевантності рекомендацій для їх усунення порівняно з традиційними підходами.

Abstract. The article presents an approach to developing an intelligent recommendation system that integrates personal computer fault diagnosis with user interaction analysis. C#, MAUI, ML.NET and EF Core are used for technical diagnostics module, while Python, Django and PostgerSQL support the analytical subsystem. The proposed architecture combines technical monitoring data with behavioral analysis to generate personalized recommendations. This integration is expected to improve the accuracy of fault prediction and the relevance of recommendations compared to traditional solutions.

Keywords: recommendation system, machine learning, diagnostics, user interaction, microservices.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Евристична оптимізація розстановки контрольних точок в технологічних процесах при багатовимірному просторі типів дефектів / О. П. Ротштейн, С. Д. Штовба, С. Б. Дубіненко, О. М. Козаченко. *Вісник Вінницького політехнічного інституту.* 2025. № 4(181). DOI: 10.31649/1997-9266-2025-181-4-85-89.

2. Штовба С. Д., Штовба О. В., Панкевич О. Д. Критерії точності та компактності для оцінювання якості нечітких баз знань в задачах ідентифікації. *Наукові праці Вінницького національного технічного університету.* 2012. № 4. URL: http://praci.vntu.edu.ua/index.php/praci/article/view/343

3. Antonov Y., Smoktii K. Expert systems using for answers analysis in automated knowledge control systems. *Herald of Khmelnytskyi National University. Technical Sciences.* 2024. Vol. 339, № 4. P. 323–331. DOI: 10.31891/2307-5732-2024-339-4-51.

4. Michael Collins. Network Security Through Data Analysis. O'Reilly, 2016. 308 p. URL: https://sar.ac.id/stmik_ebook/prog_file_file/EcuTHbNQtg.pdf

5. Jannach D., Abdollahpouri H. A survey on multi-objective recommender systems. Front Big Data, 2023. 12 p.
6. Fernandes M., Corchado J. M., Marreiros G. Machine learning techniques applied to mechanical fault diagnosis and fault prognosis in the context of real industrial manufacturing use-cases: a systematic literature review. *SPRINGER NATURE Link.* 2022. Vol. 52. P. 14246–14280. DOI: 10.1007/s10489-022-03344-3.
7. Zachariades C., Xavier V. A Review of Artificial Intelligence Techniques in Fault Diagnosis of Electric Machines. *MDPI.* 2025. № 25(16). 22 p. DOI: https://doi.org/10.3390/s25165128.
8. Ashley Davis. Bootstrapping Microservices, Second Edition: With Docker, Kubernetes, GitHub Actions, and Terraform. *Dokumen.PUB.* 2024. 464 p. URL: https://surl.li/walbsd
9. Wes McKinney. Python for Data Analysis: Data Wrangling with pandas, NumPy and Jupyter. O'Reilly, 2022. 579 p.
10. Oggl B., Kofler M. Docker: Practical Guide for Developers and DevOps Teams – Unlock the Power of Containerization: Skills for Building, Securing, and Orchestrating with Docker. Rheinwerk Computing, 2023. 491 p.
11. Esposito D., Esposito F. Programming ML.NET (Developer Reference). Microsoft, 2022. 256 p. URL: https://dl.ebooksworld.ir/books/Programming.ML.NET.9780137383658.EBooksWorld.ir.pdf
12. Trevoir Williams. Microservices Design Patterns in .NET: Making sense of microservices design and architecture using .NET Core. 2023. 300 p.