Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Кваліфікаційна наукова
праця на правах рукопису

**Чимшир Вячеслав Іванович**

УДК 004.65

# ДИСЕРТАЦІЯ

## Моделі, методи та платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг

126 – Інформаційні системи та технології
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
_____ В.І. Чимшир

Науковий керівник: Теленик С. Ф. , д.т.н., професор

Київ – 2026

# АНОТАЦІЯ

*Чимшир В. І.* Моделі, методи та платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 126 – Інформаційні системи та технології в галузі знань 12 – Інформаційні технології. – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ, 2026.

Дисертаційна робота присвячена вирішенню задачі підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг. Нові форми співпраці ІТ-компаній і провайдерів, динаміка функціональних і нефункціональних вимог обумовлюють потребу у платформі підтримки життєвого циклу сервісів в інформаційних та хмарних системах зазначених провайдерів.

Існує багато інформаційних технологій та інструментів для вирішення проблем провайдерів інформаційно-комунікаційних послуг, але відсутні платформні рішення для підтримки життєвого циклу цього класу систем як сервісу. Відповідно вимагають розроблення математичні моделі, методи та засоби моделювання управління комплексом процесів та ефективні моделі і методи для кожного процесу. Постала потреба породжує актуальну науково-практичну задачу розроблення моделей, методів і створення на їх основі платформи підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг.

Автоматизація робіт зі створення інформаційних систем провайдерів інформаційно-комунікаційних послуг за допомогою зазначеної платформи дозволить скоротити терміни їх розроблення і реалізації, зменшити витрати на їх впровадження, операційну підтримку і розвиток, забезпечити інтелектуальний моніторинг їх функціонування, поліпшити їх ефективність, продуктивність, відмовостійкість, розширюваність та інші якісні показники.

Виконано аналіз проблеми управління життєвим циклом сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг, вивчено сучасний стан реалізації сервісних платформ, на основі чого визначено завдання створення теоретичних основ підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг і основні вимоги до платформи підтримки життєвого циклу сервісів.

Запропоновано архітектуру платформи підтримки життєвого циклу сервісів, побудовану на основі мікросервісної архітектури, в інформаційних системах провайдерів інформаційно-комунікаційних послуг, яка базується на моделі End-to-End та використовує поєднання моделей штучного інтелекту, математичного програмування і теорії прийняття рішень для підтримки процесів проєктування, реалізації, експлуатації та надання сервісів, що дозволяє скоротити терміни впровадження інформаційних систем, зменшити витрати на їх розгортання та операційну підтримку, а також підвищити показники продуктивності, відмовостійкості, придатності та розширюваності.

Запропоновано метод формування архітектури сервісів в інформаційних системах провайдерів ІКП, який відрізняється структурою і набором елементів нейромережевої моделі з використанням методів глибокого навчання типу трансформер і підходом до її навчання, що дозволило використовувати її як інструмент синтезу архітектурних рішень для сервісів, які забезпечують ефективне використання, прогнозування і управління ресурсами з урахуванням параметрів сервісів, вимог провайдерів та характеристик множини їх клієнтів.

Розроблено комбінований метод групування сервісів у пакети, який відрізняється поєднанням декомпозиції задачі на підзадачі ІТ-компанії і провайдерів інформаційно-комунікаційних послуг, ймовірнісно-жадібного алгоритму і алгоритму мурашиної колонії для розв’язання підзадач, евристичного пошуку компромісу, що дозволяє групувати сервіси у взаємовигідні пакети в різних умовах з урахуванням системи знижок до преференційної ціни, специфічних особливостей провайдера і його клієнтів, ресурсів ІТ-компанії, сучасних концепцій ведення бізнесу, накопиченого

досвіду, гнучко регулювати баланс між дослідженням нових комбінацій і посиленням вже успішних розв’язків, підтримуючи самоорганізоване й адаптивне навчання.

Модифіковано керований генетичний алгоритм, який відрізняється додатковими параметрами двостороннього тиску відбору і відносної швидкості наближення до оптимуму та правилом знижок, введеними для поєднання відомої ідеї управління збіжністю алгоритму і нової концепції відбору розв’язків з врахуванням багатокритеріальності, що забезпечує досягнення глобального оптимуму при розв’язанні задачі групування сервісів у пакети з цільовими функціями для ІТ-компанії і провайдерів за рахунок зростання значень обох функцій або компенсації зменшення значення однієї функції більшим зростанням іншої, що дозволило отримувати близькі до оптимальних рішення за умови виконання обмежень.

Отримала подальший розвиток архітектура інтелектуального асистента для організації і підтримки процесів життєвого циклу сервісів як інтегральної основи розуміння і розв’язання проблем бізнес-діяльності ІТ-компанії і провайдерів і генерації відповідей з використанням комплексу великих мовних моделей і RAG-систем, які забезпечують взаємодію компонентів платформи підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг в умовах природномовної взаємодії з виконавцями в рамках проблемного контексту з використанням накопичених текстових, табличних і графічних даних та методів інтелектуального аналізу даних за схемою взаємодії інтелектуальних агентів у складі мультиагентної системи для підключення інструментів платформи для розв’язання проблем користувачів.

Створена платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг і виконане її експериментальне дослідження. Використання платформи дозволило скоротити терміни впровадження інформаційних систем, зменшити витрати на їх впровадження та операційну підтримку і поліпшити їх продуктивність,

відмовостійкість, придатність, розширюваність та інші якісні показники за рахунок комплексної автоматизації усіх процесів життєвого циклу сервісів.

Розроблені моделі і методи та створена на їх основі платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг можуть бути використані для підтримки процесів життєвого циклу сервісів в інформаційних системах провайдерів.

**Ключові слова:** інформаційна технологія, хмарні системи, мікросервісна архітектура, нейромережа, велика мовна модель, мультиагентні системи, інтелектуальний аналіз, інтелектуальний моніторинг, моделювання, машинне навчання, прогнозування, формальна модель, штучний інтелект, управління ресурсами, групування сервісів.

## Список публікацій здобувача

*Наукові праці, в яких опубліковано основні наукові результати дисертації*:

1. Чимшир В., Теленик С., Ролік О., Жаріков Е. Платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг // *Адаптивні системи автоматичного управління*. 2023. Vol. 1, No. 42. DOI: 10.20535/1560-8956.42.2023.279172.

Особистий внесок Чимшира В.: платформа підтримки життєвого циклу сервісів. Особистий внесок Теленика С.: перевірка роботи, редагування. Особистий внесок Роліка О.: аналіз процесів життєвого циклу сервісів у галузі інфокомунікацій, які потребують підтримки. Особистий внесок Жарікова Е.: аналіз специфіки впровадження сервісного підходу у галузі інфокомунікацій.

2. Гавриленко О., Чимшир В., Жаріков Е., Теленик С., Омельченко Р. Вирішення задачі впровадження пакетів сервісів за допомогою статистичної інформації // *Адаптивні системи автоматичного управління*. 2023. Vol. 2, No. 43. P. 94–107. DOI: 10.20535/1560-8956.43.2023.292257.

Особистий внесок Чимшира В.: перша формальна постановка задачі групування сервісів, підхід до її розв'язання за допомогою статистичного методу. Особистий внесок Гавриленко О.: підбір вхідних даних для експериментального дослідження. Особистий внесок Жарікова Е.: участь у експериментальному дослідженні. Особистий внесок Теленика С.: перевірка роботи, редагування. Особистий внесок Омельченка Р.: програмна реалізація.

3. Гавриленко О., Чимшир В., Жаріков Е., Теленик С., Амонс О. Метод формування пакетів сервісів за допомогою алгоритмів кластеризації // *Адаптивні системи автоматичного управління*. 2024. Vol. 1, No. 44. P. 182–191. DOI: 10.20535/1560-8956.44.2024.302437.

Особистий внесок Чимшира В.: друга формальна постановка задачі групування сервісів у пакети і підхід до її розв'язання за допомогою алгоритмів кластеризації. Особистий внесок Гавриленко О.: підбір вхідних даних для експериментального дослідження. Особистий внесок Жарікова Е.: участь у експериментальному дослідженні. Особистий внесок Амонса О.: участь у програмній реалізації. Особистий внесок Теленика С.: перевірка роботи.

4. Chymshyr V., Zhdanova O., Havrylenko O., Nowakowski G., Telenyk S. Models and methods for forming service packages for solving the problem of

designing services in information systems of providers // *Information, Computing and Intelligent Systems Journal*. 2024. No. 5. P. 29–54. DOI: 10.20535/2786-8729.5.2024.316432.

Особистий внесок Chymshyr V.: третя формальна постановка задачі групування сервісів у пакети і варіант керованого генетичного алгоритму. Особистий внесок Zhdanova O.: аналіз придатності існуючих алгоритмів для реалізації двоетапного методу. Особистий внесок Havrylenko O.: аналіз існуючих підходів до підтримки процесів етапу ініціації сервісів. Особистий внесок Nowakowski G.: участь у програмній реалізації і експериментальних дослідженнях. Особистий внесок Telenyk S.: перевірка роботи, редагування.

5. Шимкович В., Чимшир В., Знова К., Ювженко Д., Новаковський Г., Теленик С. Синтез архітектури сервісів на основі нейромережі з глибоким навчанням // *Адаптивні системи автоматичного управління*. 2025. Vol. 2, No. 47. P. 231–249. DOI: 10.20535/1560-8956.47.2025.340231.

Особистий внесок Чимшира В.: архітектура нейромережі побудови архітектури сервісів і технологія її навчання. Особистий внесок Шимковича В.: аналіз нейромереж типу трансформер. Особистий внесок Знови К.: підбір наборів даних для навчання нейромережі. Особистий внесок Ювженка Д.: участь у навчанні нейромережі. Особистий внесок Новаковського Г.: аналіз підходів до побудови архітектур сервісів. Особистий внесок Теленика С.: перевірка роботи, редагування.

6. Yuvzhenko D., Chymshyr V., Shymkovych V., Znova K., Nowakowski G., Telenyk S. A multimodal retrieval-augmented generation system with ReAct agent logic for multi-hop reasoning // *Information, Computing and Intelligent Systems Journal*. 2025. No. 6. P. 42–57. DOI: 10.20535/2786-8729.6.2025.330777.

Особистий внесок Chymshyr V.: структура і механізми функціонування мультимодальної RAG-системи з урахуванням потреб реалізації проблемного підходу. Особистий внесок Yuvzhenko D.: аналіз сучасних RAG-систем. Особистий внесок Shymkovych V.: участь у реалізації. Особистий внесок Znova K.: підбір текстових, табличних і графічних даних для навчання LLM. Особистий внесок Nowakowski G.: участь у реалізації і проведенні експериментальних досліджень RAG-системи. Особистий внесок Telenyk S.: перевірка роботи, редагування.

7. Жданова О., Букасов М., Цимбал С., Чимшир В., Омельченко Р. Інформаційна система формування пакетів сервісів для провайдерів інфокомунікацій // *Computer Systems and Information Technologies*. 2025. No. 3. P. 99–114. DOI: 10.31891/csit-2025-3-10.

Особистий внесок Чимшира В.: комбінований метод групування сервісів у пакети. Особистий внесок Жданової О.: аналіз придатності існуючих алгоритмів для реалізації етапів комбінованого методу. Особистий внесок Букасова М.: перевірка роботи, редагування. Особистий внесок Цимбала С.: експериментальне дослідження. Особистий внесок Омельченка Р.: участь у програмній реалізації методу.

8. Чимшир В. Інтелектуальний двигун і моделі та методи платформи комплексної підтримки процесів життєвого циклу інфокомунікаційних сервісів // *Технічні науки та технології*. 2025. No. 3(41). P. 284–295. DOI: 10.25140/2411-5363-2025-3(41)-284-295.

Особистий внесок Чимшира В.: концепція інтелектуального асистента платформи підтримки процесів життєвого циклу сервісів і її реалізація.

9. Чимшир В., Омельченко Р. Вибір методу формування портфеля сервісів для провайдерів інфокомунікацій з урахуванням ситуації // *Наукові праці ВНТУ*. 2025. No. 3. P. 1–12. DOI: 10.31649/2307-5376-2025-3-149-160.

Особистий внесок Чимшира В.: мікросервісна архітектура інформаційної системи групування сервісів у пакети і підхід до врахування опіній бізнес- і IT-менеджерів для вибору найбільш придатного методу в залежності від ситуації. Особистий внесок Омельченка Р.: участь у програмній реалізації методів і проведенні експериментальних досліджень.

*Наукові праці, які засвідчують апробацію матеріалів дисертації:*

10. Chimshir Vyacheslav. Approach to design and development of services in information systems of telecommunications providers / Vyacheslav Chimshir, Olena Gavrylenko, Grzegorz Nowakowski, Eduard Zharikov, Sergii Telenyk // Proc. of the International Conference on Security, Fault Tolerance, Intelligence. – Київ, 2023. – С. 1–5.

Особистий внесок Chymshyr V.: підхід до побудови архітектури і реалізації сервісів. Особистий внесок Havrylenko O.: аналіз підходів до групування сервісів у пакети. Особистий внесок Nowakowski G.: підхід до ведення каталогу сервісів.

Особистий внесок Zharikov E.: аналіз впливу соціальних мереж. Особистий внесок Telenyk S.: перевірка роботи, редагування.

11. Теленик С. Дослідження процесів, пов’язаних з проектуванням сервісів у інформаційних системах провайдерів / С. Ф. Теленик, Гжегош Новаковський, О. В. Гавриленко, В. І. Чимшир, Р. В. Омельченко // VI Міжнародна науково-практична конференція молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології (SOFT TECH-2024)»: тези доповідей міжнар. конф. – Київ, 2024. – С. 140–148.

Особистий внесок Чимшира В.: підхід до побудови архітектури сервісів. Особистий внесок Теленика С.: перевірка роботи, редагування. Особистий внесок Новаковського Г.: тестування і валідація сервісів. Особистий внесок Гавриленко О.: функціональні вимоги. Особистий внесок Омельченка Р.: графічний інтерфейс сервісу.

12. Ювженко Д. І. Автоматизація бізнес-процесів за допомогою мульти-агентної ШІ-системи / Д.І. Ювженко, В.І. Чимшир, В. М. Шимкович, С.Ф. Теленик // Матеріали міжнародної наукової інтернет конференції “Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 97), (м. Тернопіль, Україна, м. Ополе, Польща, 13-14 березня 2025 р.)”. – Тернопіль : ФО-П Шпак В.Б. 2025. 102 с. – ISSN 2522-932X. – С. 40–44.

Особистий внесок Чимшира В.: концепція агентної системи і її прототип. Особистий внесок Ювженка Д.: підбір фреймворка для реалізації. Особистий внесок Шимковича В.: участь у програмній реалізації. Особистий внесок Теленика С.: перевірка роботи, редагування.

13. Жданова О. Модель і метод формування пакетів сервісів для провайдерів інфокомунікацій / О. Жданова, М. Букасов, С. Цимбал, В. Чимшир, Р. Омельченко // XIII Міжнародна науково-практична конференція InfoCom Advanced Solutions 2025, 21–22 травня 2025 р., Київ, Україна. – К., 2025. – С. 1-6.

Особистий внесок Чимшира В.: модель і метод групування сервісів. Особистий внесок Жданової О.: підбір даних для експериментального дослідження. Особистий внесок Цимбала С.: експериментальне дослідження. Особистий внесок Букасова М.: перевірка роботи, редагування. Особистий внесок Омельченка Р.: програмна реалізація.

## ABSTRACT

Chymshyr V. I. Models, Methods and a Platform for Supporting the Service Lifecycle in Information Systems of Information and Communication Service Providers. – Qualification Scientific Work in the Form of a Manuscript.

Dissertation submitted for the degree of Doctor of Philosophy in specialty 126 – Information Systems and Technologies in the field of knowledge 12 – Information Technologies. – Igor Sikorsky Kyiv Polytechnic Institute, National Technical University of Ukraine, Kyiv, 2026.

The dissertation is devoted to solving the problem of supporting the lifecycle of services in information systems of information and communication service providers. New forms of cooperation between IT companies and information and communication service providers, as well as the dynamics of functional and non-functional requirements, determine the need for a platform supporting the full range of service lifecycle processes in the information and cloud systems of such providers.

There are many tools designed to solve the problems of information and communication service providers; however, platform solutions for supporting the lifecycle of this class of systems as a service are lacking. Accordingly, mathematical models and methods for managing a complex set of processes, as well as efficient models, methods and modeling technique for each individual process, require development. This need gives rise to an actual scientific and practical task of developing models and methods and creating, on their basis, a platform for supporting the lifecycle of services in the information systems of information and communication service providers.

Automation of the work related to the creation of information systems for information and communication service providers using the proposed platform will reduce development and implementation time, decrease the cost of deployment, operational support, and further development, and improve efficiency, performance, fault tolerance, scalability, security, intelligent monitoring and other quality indicators.

An analysis of the problem of managing the full range of service lifecycle processes in the information systems of information and communication service providers has been carried out. The current state of implementation of service platforms has been studied, and based on this analysis the tasks of creating theoretical foundations for supporting the service lifecycle processes in the information systems of information and communication service providers, as well as the main requirements for a service lifecycle processes support platform, have been defined.

The platform for supporting the service lifecycle in the information systems of infocommunication service providers have been developed and experimentally investigated. It is based on the End-to-End model and uses a combination of artificial intelligence models, mathematical programming, and decision theory to support the processes of service design, implementation, operation, and delivery. This approach reduces the time required for information system deployment, decreases the cost of deployment and operational support, and improves performance, fault tolerance, maintainability, and scalability indicators.

A method for forming service architecture in the information systems of information and communication service providers is proposed. It differs in the structure and set of elements of deep learning neural network model of the transformer type and in the approach to its training. This made it possible to use the model as a tool for synthesizing architectural solutions for services that ensure efficient use, forecasting and resource management while taking into account service functional and non-functional parameters, provider requirements, and the characteristics of their client base.

A combined method for grouping of services into packages has been developed. It differs by combining the decomposition of the problem into subproblems for IT companies and information and communication service providers, a probabilistic greedy algorithm, and an ant colony optimization algorithm for solving subproblems, as well as a heuristic search for compromise solutions. This approach makes it possible to form mutually beneficial service

packages under various conditions, taking into account discount systems for preferential pricing, specific characteristics of the provider and its customers, the resources of the IT company, modern business concepts, and accumulated experience, while flexibly balancing the exploration of new combinations and the reinforcement of successful solutions, supporting self-organized and adaptive learning.

A modified guided genetic algorithm has been developed. It differs by introducing additional parameters of bidirectional selection pressure and relative convergence rate, as well as a discount rule. These elements combine the known idea of controlling algorithm convergence with a new concept of selecting solutions under multicriteria conditions. This ensures achieving the global optimum when solving the problem of grouping of services into packages with objective functions for both the IT company and the provider by increasing the values of both functions or compensating for a decrease in one function by a greater increase in the other. This allows obtaining near-optimal solutions while satisfying the imposed constraints.

The architecture of an intelligent assistant for organizing and supporting service lifecycle processes has been further developed. It serves as an integral basis for understanding and solving business problems of IT companies and providers and for generating responses using a set of large language models and RAG systems. These components ensure interaction between the elements of the service lifecycle support platform in the information systems of providers in conditions of natural language interaction with executors within the problem context, using accumulated textual, tabular, and graphical data along with the intelligent analysis according to an intelligent agent as part of multi-agent systems for connecting platform tools to solve user problems.

The developed models and methods, as well as the platform created on their basis for supporting the lifecycle of services in the information systems of information and communication service providers, can be used to support service lifecycle processes in provider information systems.

**Keywords:** information technology, cloud systems, microservice architecture, neural network, large language model, multi-agent systems, intelligent analysis, intelligent monitoring, modeling, machine learning, forecasting, formal model, artificial intelligence, resource management, grouping of services.

ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ЖЦ – життєвий цикл

ІКП – інформаційно-комунікаційні послуги

ІС – інформаційна система

ІТ – інформаційна технологія

ОПР – особа, що приймає рішення

СКБД – система керування базами даних

СОА – сервіс-орієнтована архітектура

ШІ – штучний інтелект

API – Application Programming Interface – прикладний програмний інтерфейс

BSS – Business Support System – система підтримки бізнесу

CI/CD – Continuous Integration/Continuous Delivery – неперервна інтеграція/неперервна доставка

CMMI – Capability Maturity Model Integration – інтеграція моделі зрілості можливостей

GUI – Graphic User Interface – графічний інтерфейс користувача

HTTP – Hyper-Text Transfer Protocol – протокол передачі гіпертексту

IaaS – Infrastructure as a Service – інфраструктура як сервіс

ITIL – Information Technology Library – бібліотека інформаційних технологій

ITSM – Information Technology Service Management – управління послугами інформаційних технологій

LLM – Large Language Model – велика мовна модель

OSS – Operation Support System – система підтримки операцій

PaaS – Platform as a Service – платформа як сервіс

QoS – Quality of Service – якість послуг

RAG – Retrieval-Augmented Generation – генерація з доповненим пошуком

SDPac – Service Design Package – пакет проєктування сервісів

SLA – Service-Level Agreement – угода про рівень послуг

SaaS – Software as a Service – програмне забезпечення як сервіс

# ВСТУП

**Актуальність теми.** Сучасний ринок інформаційно-комунікаційних послуг (ІКП) характеризується переходом до сервісно-орієнтованих моделей співпраці, у яких ІТ-компанії проєктують, реалізують і забезпечують експлуатаційну підтримку інформаційних систем (ІС) провайдерів ІКП на основі наскрізної – End-to-End (E2E) відповідальності за результат. Такі умови супроводжуються високою динамікою функціональних і нефункціональних вимог, скороченням термінів виведення сервісів на ринок, зростанням залежностей між сервісами та жорсткішими вимогами до показників якості (продуктивність, відмовостійкість, придатність, розширюваність, дотримання SLA та ін.). Це зумовлює потребу у створенні платформи підтримки життєвого циклу (ЖЦ) сервісів, здатної забезпечити цілісну підтримку процесів проєктування, реалізації, передачі до надання, надання та розвитку сервісів в ІС провайдерів ІКП.

Питанням створення зазначеної платформи підтримки ЖЦ сервісів, здатної автоматизувати комплексну підтримку усіх процесів і скласти основу створення ІС провайдерів ІКП, які гарантують дотримання наведених вище вимог, присвячено ряд досліджень як українських, так і іноземних вчених [1–20].

Незважаючи на наявність ряду рішень для автоматизації окремих етапів ЖЦ (керування вимогами, розвиток, моніторинг, ITSM/ESM, керування інцидентами тощо), у практиці E2E-моделі взаємодії залишається недостатньо вирішеною задача інтегрованого управління ЖЦ сервісів як єдиної системи, у якій рішення на ранніх етапах проєктування узгоджуються з подальшим наданням і розвитком сервісів та з ресурсними обмеженнями ІТ-компанії і провайдера. Додатково ускладнює ситуацію необхідність підтримки діяльності виконавців у процесах ЖЦ в умовах роботи з різнорідними накопиченими даними (текстовими, табличними, графічними) та потреба у природномовній взаємодії з інструментами платформи для швидкого виявлення проблем, формування рішень і запуску відповідних процедур.

Отже, на даний час спостерігається суперечність між: 1) потребою IT-компаній і провайдерів ІКП у цілісній платформі підтримки ЖЦ сервісів в ІС в умовах E2E-моделі співпраці, яка забезпечує узгодженість рішень за всіма етапами ЖЦ, скорочення термінів упровадження сервісів, зниження витрат та забезпечення показників якості й SLA та 2) недостатньою розвиненістю моделей, методів і інформаційних технологій (ІТ), що забезпечують інтегроване проєктування сервісів, групування сервісів у пакети і організацію виконання бізнес-процесів ЖЦ із використанням інтелектуальних засобів природномовної взаємодії та оптимізації ресурсів.

Зазначена суперечність визначає актуальну науково-технічну задачу створення платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП на основі E2E-моделі, що вирішується у даній дисертаційній роботі шляхом аналізу провайдерів ІКП, уточнення задачі та розроблення комплексу моделей, методів та ІТ підтримки процесів ЖЦ сервісів.

**Зв'язок роботи з науковими програмами, планами, темами.** Дослідження за темою дисертаційної роботи провадилось у Національному технічному університеті України «Київський політехнічний інститут імені Ігоря Сікорського» в рамках виконання ініціативної науково-дослідницької роботи «Інтелектуальні високопродуктивні технології управління технічними системами» (номер державної реєстрації – 0121U110810).

**Мета і задачі дослідження.** *Об'єкт дослідження* – процеси підтримки ЖЦ сервісів в ІС провайдерів ІКП, які проектуються, реалізуються, підтримуються і розвиваються на основі моделі E2E.

*Предмет дослідження* – моделі, методи та ІТ підтримки ЖЦ сервісів в ІС провайдерів ІКП на основі E2E-моделі, включно з технологіями інтелектуальної підтримки бізнес-процесів, проєктування сервісів і групування сервісів у пакети.

*Метою* дисертаційної роботи є скорочення термінів розроблення та впровадження ІС провайдерів ІКП, зниження витрат на їх розроблення, впровадження та операційну підтримку, а також покращення показників

якості (продуктивності, відмовостійкості, придатності, розширюваності тощо) шляхом створення платформи підтримки ЖЦ сервісів на основі E2E-моделі та розроблення відповідних моделей, методів і ІТ підтримки процесів ЖЦ сервісів.

Для досягнення поставленої мети розв'язано такі основні *задачі*:

– виконати аналіз сучасного стану, моделей і методів підтримки ЖЦ сервісів та підходів до проєктування сервісів і групування сервісів у пакети в ІС провайдерів ІКП;

– обґрунтувати вимоги та принципи побудови платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП на основі E2E-моделі та розробити її архітектуру;

– розробити ІТ природномовної взаємодії з компонентами платформи та підтримки виконання бізнес-процесів діяльності на основі LLM- і RAG-орієнтованого підходу з урахуванням контексту проблеми виконавця та різнорідних даних;

– запропонувати метод формування архітектури сервісів в ІС провайдерів ІКП з урахуванням параметрів сервісів, вимог провайдера та характеристик клієнтів, орієнтований на ефективне використання ресурсів (із використанням нейромережевих моделей як інструменту побудови рішень);

– розробити комплекс моделей і методів групування сервісів у пакети в ІС провайдерів ІКП на основі задачі змішаного програмування з використанням статистичних методів, алгоритмів кластеризації, методів математичного програмування, теорії прийняття рішень та еволюційних методів оптимізації;

– розробити комбінований метод і удосконалити керований генетичний алгоритм багатокритеріальної оптимізації пакетів сервісів для досягнення взаємної вигоди ІТ-компанії та провайдерів з урахуванням SLA, ресурсних обмежень, міжсервісних залежностей і правил знижок;

– реалізувати програмний прототип платформи підтримки ЖЦ сервісів, провести дослідження її функціональних можливостей та оцінювання ефективності використання отриманих результатів.

**Методи дослідження:** У роботі використано методи системного аналізу та теорії ІС, методи дослідження операцій і математичного програмування, теорії ймовірностей і математичної статистики, методи оптимізації та еволюційні алгоритми, методи штучного інтелекту та інтелектуального аналізу даних, моделювання бізнес-процесів і підтримки прийняття рішень.

**Наукова новизна дисертаційного дослідження** полягає у отриманні таких наукових результатів:

1) уперше запропоновано архітектуру платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП, яка базується на моделі End-to-End та використовує поєднання моделей штучного інтелекту, математичного програмування і теорії прийняття рішень для підтримки процесів проєктування, реалізації, експлуатації та надання сервісів, що дозволяє скоротити терміни впровадження ІС, зменшити витрати на їх розгортання та операційну підтримку, а також підвищити показники продуктивності, відмовостійкості, придатності та розширюваності;

2) уперше запропоновано метод формування архітектури сервісів в ІС провайдерів ІКП, який відрізняється структурою і набором елементів нейромережевої моделі з глибоким навчанням типу трансформер і підходом до її навчання, що дозволило використовувати її як інструмент синтезу архітектурних рішень для сервісів, які забезпечують ефективне використання та управління ресурсів з урахуванням параметрів сервісів, вимог провайдерів та характеристик множини їх клієнтів;

3) уперше розроблено комбінований метод групування сервісів у пакети, який відрізняється поєднанням декомпозиції задачі на підзадачі ІТ-компанії і провайдерів ІКП, ймовірнісно-жадібного алгоритму і алгоритму мурашиної колонії для розв’язання підзадач, евристичного пошуку компромісу, що дозволяє формувати взаємовигідні пакети сервісів в різних умовах з урахуванням системи знижок до преференційної ціни, специфічних особливостей провайдера і його клієнтів, ресурсів ІТ-компанії, сучасних концепцій ведення бізнесу, накопиченого досвіду, гнучко регулювати баланс

між дослідженням нових комбінацій і посиленням вже успішних розв’язків, підтримуючи самоорганізоване й адаптивне навчання;

4) модифіковано керований генетичний алгоритм, який відрізняється додатковими параметрами двостороннього тиску відбору і відносної швидкості наближення до оптимуму та правилом знижок, введеними для поєднання відомої ідеї управління збіжністю алгоритму і нової концепції відбору розв’язків з врахуванням багатокритеріальності, що забезпечує досягнення глобального оптимуму при розв’язанні задачі групування сервісів у пакети з цільовими функціями для ІТ-компанії і провайдерів за рахунок зростання значень обох функцій або компенсації зменшення значення однієї функції більшим зростанням іншої, що дозволило отримувати близькі до оптимальних рішення за умови виконання обмежень;

5) отримала подальший розвиток архітектура інтелектуального асистента для організації і підтримки процесів ЖЦ сервісів як інтегральної основи розуміння і розв’язання проблем бізнес-діяльності ІТ-компанії і провайдерів і генерації відповідей з використанням комплексу великих мовних моделей і RAG-систем, які забезпечують взаємодію компонентів платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП в умовах природномовної взаємодії з виконавцями в рамках проблемного контексту з використанням накопичених текстових, табличних і графічних даних за схемою взаємодії  інтелектуальних агентів для підключення інструментів платформи для розв’язання проблем користувачів.

**Практичне значення отриманих результатів**. Практичне значення полягає у можливості використання розробленої платформи підтримки ЖЦ сервісів та її компонентів у діяльності ІТ-компаній, які створюють і супроводжують ІС провайдерів ІКП за E2E-моделлю. Запропонована інформаційна технологія природномовної взаємодії на основі LLM і RAG забезпечує підтримку виконання бізнес-процесів і підключення інструментів платформи відповідно до контексту задачі, що підвищує оперативність вирішення інцидентів і узгодженість дій виконавців.

Комплекс моделей і методів проєктування сервісів та групування сервісів у пакети забезпечує прийняття обґрунтованих рішень з урахуванням ресурсних обмежень, параметрів сервісів, характеристик клієнтів, вимог SLA та міжсервісних залежностей.

Результати дисертаційної роботи впроваджено у (Додаток А): навчальний процес кафедри Інформаційних систем та технологій факультету Інформатики та обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського».

**Апробація матеріалів дисертації**. Основні положення та результати дисертації доповідались і обговорювались на науково-практичних конференціях і семінарах:

– Міжнародна конференція “Security, Fault Tolerance, Intelligence”, Київ, 29-30 липня, 2023;

– VI Міжнародна науково-практична конференція молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології (SOFT TECH-2024)», Київ, 20-22 листопада, 2024;

– Міжнародна науково-практична інтернет-конференція «Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 97)», Харків, 13-14 березня, 2025;

– XIII Міжнародна науково-практична конференція InfoCom Advanced Solutions, Київ, 21–22 травня, 2025.

**Публікації.** За результатами дисертаційного дослідження опубліковано 9 наукових праць, серед яких праці [68–73, 142–144], наведені у списку використаних джерел.

**Структура та обсяг роботи.**

Дисертаційна робота складається із вступу, чотирьох розділів, висновків, списку використаних джерел (144 найменувань) і шести додатків. Основний зміст викладений на 162 сторінках друкованого тексту, містить 14 рисунків та 4 таблиці. Загальний обсяг дисертації – 241 сторінок.

# 1 АНАЛІЗ ПРОБЛЕМИ ПІДТРИМКИ ЖИТТЄВОГО ЦИКЛУ СЕРВІСІВ В ІНФОРМАЦІЙНИХ СИСТЕМАХ ПРОВАЙДЕРІВ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ПОСЛУГ

У розділі виконано аналіз сервісно-орієнтованого підходу, підходів до реалізації сервісної платформи у різних прикладних галузях і запропонована концепція платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП. Розв'язання проблеми підтримки ЖЦ сервісів розпочнемо з його аналізу, щоб зрозуміти природу ЖЦ, призначення, структуру, виявити особливості складових і їх взаємодії. Результати аналізу визначать концепцію і реалізацію загального рішення, особливості математичних моделей і методів підтримки ЖЦ сервісів.

## 1.1 Життєвий цикл сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг та проблема його підтримки

Сьогодні сервісний підхід широко впроваджується у всі галузі людської діяльності [1, 2]. Цей перехід, який отримав назву "servitization" [3], притаманний як сервісним компаніям, так і несервісним [4]. Він супроводжується процесом digitalization [2, 5] в рамках загального процесу цифрової трансформації [6]. За рахунок надання інноваційних цифрових сервісів бізнес може створювати і отримувати нові цінності [7]. Цифрова трансформація підвищує ефективність і продуктивність бізнесу, але її впровадження вимагає відповідних технологій [8].

Сервісний підхід змінив характер співпраці між ІТ-підрозділами та бізнес-підрозділами. Акцент робиться на підтримці бізнес-процесів підприємств, компаній, бізнес вимагає готових рішень для вирішення своїх проблем, спілкування відбувається в термінах цілей, проблем, можливостей, переваг, рішень [9–11]. Зростає роль ІТ-сервісів у підтримці бізнес-процесів з метою проєктування, реалізації, надання і розвитку сервісів.

У цих умовах ІС провайдерів ІКП і надавців сервісів в інших галузях будуються на основі концепції ЖЦ сервісів [12]. Виникає потреба в моделях, методах і технологіях відстеження тенденцій галузі, прогнозування ринку,

оцінювання параметрів SLA, забезпечення рівня QoS, інтелектуального моніторингу стану сервісів, безпеки і т.п. [13].

Абстрактний погляд на ЖЦ сервісів пов'язує з ним послідовність операцій, поєднаних за аспектами часу проєктування та виконання [14–16]. ЖЦ сервісів структурується – перший аспект охоплює моделювання, проєктування, реалізацію і тестування, а другий – розгортання, надання і розвиток сервісів. Ця концепція дозволяє поєднати операції, які звичайно розглядають ізольовано [17, 18]. Потреба враховувати середовище надання сервісів, управляти ресурсами вимагає структурувати сервіси, що призводить до міцнішого переплетення компонентів окремих процесів [19, 20]. Дослідження ЖЦ сервісу з урахуванням особливостей галузі інфокомунікацій буде мати суттєвий вплив на рішення.

Реалізація концепції структурування вимагає представлення ЖЦ сервісів як цілого, так і окремих його процесів. Це визначає потребу у формальних моделях і методах сервісів і процесів їх ЖЦ, які будуть покладені в основу реалізації інформаційних технологій та інструментів їх підтримки в хмарних системах і розподілених середовищах. Різнорівневе моделювання уможливить автоматизацію діяльності ІТ-компаній і провайдерів ІКП, враховуючи її комплексний характер, усі процеси з врахуванням їх взаємних впливів.

Сьогодні автоматизація діяльності є інструментом досягнення бізнес-цілей суб'єктів підприємництва. Автоматизація підпорядковує діяльність цілям підприємства згідно визначеної стратегії в рамках продуманої організації з використанням сучасних методів праці і з врахуванням інтересів усіх заінтересованих сторін. Одним із напрямів автоматизації є підтримка повсякденної діяльності працівників компаній на основі бізнес-процесів.

Згідно організації підрозділи і працівники виконують певні функції, тобто знають «що треба робити». Бізнес-процеси, ролі у виконанні яких приписані співробітникам, визначають «як треба робити», виконуючи функціональні обов'язки. Бізнес-процеси становлять взаємопов'язаний комплекс, при цьому кожний процес інтегрує множину підпроцесів, які звичайно складаються із операцій. Суб'єкти господарської діяльності часто

об’єднуються у професійні спільноти, одним із завдань яких є розроблення універсальної системи бізнес-процесів для споріднених підприємств.

Саме підтримка бізнес-процесів є основоположним рівнем автоматизації діяльності компанії. Для підтримки діяльності, представленої у формі бізнес-процесів, використовуються сервіси [21, 22], які треба визначати, проектувати, реалізовувати, надавати і розвивати. Постають потреби у оцінюванні системи сервісів, її повноти (з точки зору підтримки цілісної системи бізнес-процесів), якості, ефективності, відповідності змінним умовам бізнесової діяльності. Це дозволяє говорити про управління сервісами, що, в свою чергу, обумовлює потребу у відповідних концепціях та інструментах управління ІТ-операціями. Сьогодні зазначені концепції та інструменти можна знайти у матеріалах, орієнтованих на діяльність як ІТ-компаній, так і провайдерів ІКП.

Зростає роль ITSM, розроблюються стандартизовані набори моделей для фреймворків ITSM, насамперед ITIL, ISO/IEC 20000, MOF та FitSM, з можливістю налаштування для різних процесів [14, 24].

Матеріали ITIL, призначені підтримці ЖЦ сервісів, надали імпульс появі інструментів управління сервісами з врахуванням аспектів ведення бізнесової діяльності [25, 26]. Основу створення ефективних технологій управління ІТ-сервісами склали упорядкування, класифікація і моделювання. Ключова концепція повторюваного циклу ІТ-сервісів доповнена ефективними ідеями виділення в них етапів як груп пов’язаних процесів. Це дозволило визначити вимоги до ІТ-середовища, орієнтованого на розвиток бізнесу в умовах змінності, пов’язати управління ІТ-сервісами з інфраструктурою, приймати рішення з врахуванням цілей і стратегії компанії, ризиків і обмежень, вимог клієнтів. Щоб управління ІТ-сервісами сприяло успіхам компанії, бізнес- і ІТ-підрозділи координують співпрацю для досягнення спільних цілей відповідно до спільної стратегії задовольняючи бізнес-потреби підприємства.

Практикам ITIL теж притаманна структуризація ІТ-послуг. Виділяються етапи ЖЦ послуг, кожний з яких містить свій набір процесів. У такий спосіб враховуються традиційні і нові сфери управління ІТ-послугами. Управління

процесами і етапами виконується згідно визначених завдань на основі прийнятої певної системи принципів і правил. Зазначена система орієнтована на визначену мету діяльності, враховує взаємозв'язки процесів етапів.

Зазначена структуризація ЖЦ сервісів дозволяє впорядкувати діяльність з надання послуг. Процеси виконуються згідно певних схем, формалізація яких уможливить автоматизацію процесів. Але ITIL властиві сучасні погляди на підтримку процесів, насамперед постійне відстеження і усування проблем, неперервне вдосконалення ІТ-послуг, уявлення про нефункціональні вимоги до послуг, наприклад безперервність надання, продуктивність, ефективність та ін.

У плані ефективного виконання завдань дисертаційного дослідження аналіз практик ITIL допоможе виявити структуризацію (етапи, процеси, їх зв'язки) і корисні концепції, які можна покласти в основу технологій і інструментів підтримки ЖЦ сервісів в ІС провайдерів ІКП. Перший етап – розроблення стратегії сервісу – включає процеси, пов'язані з управлінням пакетами ІТ-сервісів, коштами і бюджетами, пов'язаними з наданням сервісів, визначенням вимог клієнтів і прогнозуванням попиту, аналізом ринку і напрацюванням ринкової стратегії. Тут корисними будуть концепції підтримки процесів ЖЦ сервісів з врахуванням цілей компанії і її стратегії на основі оцінок за певними критеріями з точки зору можливостей, необхідних для задоволення потреб бізнесу. Процеси другого етапу – проєктування сервісу – охоплюють ведення каталогу сервісів, управління SLA, потужністю, розробленням, ризиками, забезпечення безперервності бізнесу, доступності сервісів, безпеки. Проєктування сервісу розглядається як трансформація його опису (визначеної стратегії) у архітектуру і плани досягнення цілей бізнесу. Тут корисними будуть концепції оцінювання сервісу як цінності для бізнесу і прийняття рішень на основі критерію ефективності ІС відповідно до потреб бізнесу. До процесів третього етапу – переходу до надання сервісу – належать оцінювання і управління змінами, керування випусками і доставами, активами служби і конфігураціями, тестування і вимірювання результатів, вирішення проблем, планування переходу сервісу у виробництво. Тут акцент робиться на

згаданій вище концепції відстеження і усування проблем. Четвертий етап – надання сервісу – охоплює процеси управління інцидентами, подіями, відновлення після збоїв, забезпечення безперервності надання сервісів, безпеки і виконання запитів на обслуговування, запобігання повторенню проблем та інцидентів. Тут видаються корисними концепції підтримки сервісів на основі описів бізнес-процесів компанії, оцінювання параметрів процесів і сервісів, інтелектуального моніторингу сервісів і процесів, відновлення порушених сервісів для мінімізації впливу на бізнес, введені критерії ефективності і продуктивності. Процеси п’ятого етапу – постійного вдосконалення сервісу – пов’язані з переглядом, вибором і моніторингом ініціатив вдосконалення, управлінням процесом покращення сервісів.

ІТ-компанія, яка надає ІТ-сервіси клієнтам, для застосування ITIL має вибрати комплекс зазначених сервісів і підтримувати процеси їх ЖЦ. Вибір і підтримка мають здійснюватися відповідно до цілей і стратегії бізнесу, за певними критеріями оптимізації. У сучасних умовах ведення бізнесу компанія має поліпшувати рівень обслуговування клієнтів, розвивати довгострокові відносини, розвивати бізнес, враховуючи обмеження і ризики. Це вимагає ефективних методів праці в рамках стабільного ІТ-середовища, орієнтованого на розвиток, підкреслює потребу в автоматизації процесів, адаптації до змін.

Постає складна проблема підтримки процесів ЖЦ сервісів з огляду на наведені вище чинники, розміри компаній, структурованість і значні обсяги ресурсів, різноманіття технологій. Її ускладнюють жорсткі нефункціональні вимоги. Використання подібних практик вимагає відповідних інструментів. Оскільки процеси усіх етапів ЖЦ сервісів компаній специфічні, недоцільно розробляти типове рішення для кожних сервісу і процесу, прийнятне для усіх компаній. Необхідне універсальне рішення для кожного сервісу і кожного його процесу, налаштовувано на основі описів сервісу і його процесів, щоб всі процеси сервісу підтримувалися з врахуванням специфіки компанії.

Було б доцільним, щоб таке універсальне рішення можна було застосувати для підтримки усіх ІТ-сервісів і їх процесів. Саме таким рішенням є платформа,

яка проєктування, реалізацію, впровадження, надання і розвиток ІТ-сервісів виконує згідно їх описів, шаблонів, контексту на основі математичних моделей і методів, покладених в основу функціонування відповідних технологій та інструментів платформи. Така платформа може бути налаштована на процеси інших етапів, наприклад оцінювання параметрів процесів і сервісів та ін. [27, 28].

Аналіз сервісів, які надають провайдери ІКП, і технологій, якими їх для цього забезпечують ІТ-компанії, показав, що у створення такої платформи можна покласти низку ефективних концепцій, які зарекомендували себе у створенні технологій та інструментів управління процесами ЖЦ ІТ-сервісів. По-перше, для процесів етапів ініціації, аналізу, інженерії вимог можна використовувати напрацьовані моделі, методи і технології з врахуванням специфіки галузі і компанії та застосуванням концепцій бізнес-цілей, стратегій, критеріїв, обмежень і ризиків, орієнтованого на розвиток середовища в рамках розглянутих підходів до групування і взаємодії процесів.

По-друге, виходячи з специфіки галузі інфокомунікацій і особливостей провайдерів ІКП, необхідно визначити процеси для кожного етапу, надати їх описи і шаблони. Описи бізнес-процесів можна подавати графічними мовами.

По-третє, ключовий етап проектування сервісів для провайдерів ІКП полягає у трансформації контексту і опису сервісу в його архітектуру. Опис сервісу звичайно подається в термінах функціональних і нефункціональних вимог, параметрів провайдера, його діяльності, потреб і параметрів клієнтів.

По-четверте, реалізацію сервісів теж представлятимемо як трансформацію, але вже – архітектури сервісу в код. Реалізація процесів є складовою етапу реалізації і виконується на основі описів процесів.

По-п’яте, створення універсального інструменту вимагає ефективних моделей і методів. Такі інструменти створюються для усіх етапів або для кожного етапу окремо. Вони також забезпечать підтримку управління проектуванням та реалізацією сервісів (планування і розподіл завдань, призначення виконавців і термінів, контролю виконання та ін.). Бажаними були б інструменти підтримка процесів управління наданням ІТ-сервісів з

урахуванням SLA, виявлення і усунення проблем, оцінювання сервісів і рівня задоволеності клієнтів та ін., що вимагає додаткових моделей і методів.

По-шосте, додатково, використовуючи інші концепції, виявлені в процесі аналізу, зазначений інструмент міг би реалізовувати можливості, які вже стали звичними, наприклад використання засобів і інструментів третіх сторін (для зменшення часу і витрат на реалізацію процесів), управління процесами, ефективного використання ресурсів хмарних систем, та інші.

На структурування процесів впливають і інші галузеві документи, наприклад матеріали Комплексної моделі організаційної зрілості – Capability Maturity Model Integration (CMMI) [29]. Якщо IT-компанія групує процеси згідно наведеної на рис. 1.1 схеми групування процесів [30], її команді проєкту доступні діаграми, артефакти, ролі, обов'язки, інструменти, приклади результатів, інша корисна інформація. Вони доступні через корпоративне сховище, посилання на портали з описами, шаблонами та схемами процесів.

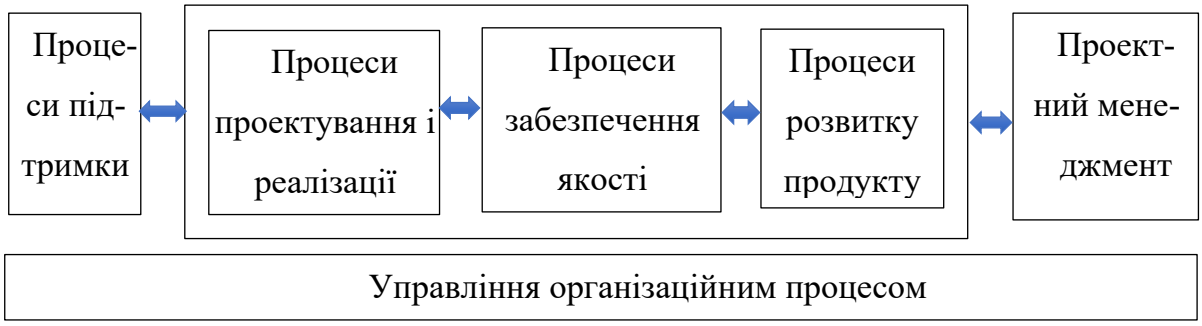


Рисунок 1.1 – Схема групування процесів IT-сервісів

Група процесів *проектного менеджменту* містить традиційні процеси управління проєктами. До групи *проєктування і реалізації* належать процеси збору та управління вимогами, проєктування архітектури рішень і системи, міграції даних, реалізація системи; інтеграція; розгортання системи; перехід до підтримки. Група процесів *забезпечення якості* поєднує контроль якості рішення, тестування обсягу напруги і аудит. Група процесів *підтримки* включає управління конфігурацією, корпоративне навчання, аналіз причин і висновки, аналіз рішень і висновки, вимірювання та аналіз, моделювання та управління продуктивністю. Група процесів *управління організаційними*

*процесами* включає управління організаційними процесами, управління регуляторними документами, вдосконалення процесів і внутрішній аудит.

Враховуючи матеріали професійних об'єднань у галузі інфокомунікацій, насамперед TM Forum [31], загальний підхід до визначення процесів підтримки традиційних етапів ЖЦ сервісів End-to-End (E2E) [32] на основі платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП в умовах сертифікації CMMI та поширення підходу Evergreen [33] і концепцій інтегрованості платформи і керованих сервісів представимо у вигляді, наведеному на рис. 1.2.


Рисунок 1.2 – Загальний підхід до підтримки ЖЦ сервісів E2E

Отже, виконаний аналіз існуючих підходів до структуризації ЖЦ сервісів в ІС провайдерів ІКП показав різноманіття процесів, які вимагають підтримки. Постає складна проблема підтримки усіх процесів ЖЦ сервісів в ІС провайдерів ІКП. Для реалізації підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП потрібні формальні моделі, методи, відповідні інструменти. Огляду існуючих інструментів підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП присвячений наступний підрозділ дисертаційної роботи.

## 1.2 Існуючі підходи до реалізації підтримки життєвого циклу сервісів

У підрозділі наведено результати аналізу існуючих інструментів підтримки процесів ЖЦ сервісів. Виходячи з завдань дисертаційного дослідження і сформульованої проблеми підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП, спочатку сконцентруємося на розгляді переваг і недоліків існуючих інструментів підтримки процесів ЖЦ сервісів у різних галузях, насамперед ІТ та телекомунікацій. Потім розглянемо і оцінимо перспективи використання для вирішення згаданої вище проблеми нових концепцій, якими останнім часом збагатилися співпраця ІТ-компаній і провайдерів ІКП. При цьому перевагу будемо надавати комплексним рішенням, передумови застосування яких поступово викристалізовуються у ході дослідження.

Розглянемо платформні рішення, комплексність яких супроводжується автоматизацією підтримки процесів усіх етапів ЖЦ сервісів, основаною на широкому використанні математичних моделей і методів, покладених в основу реалізації технологій для підтримки процесів ЖЦ сервісів. Простір рішень охоплює різнорідні платформи і середовища розроблення сервісів для кінцевих користувачів/третіх сторін. Розпочнемо з платформи B3G [34], реалізаціями якої виступають PLASTIC [35], SPICE [36], OPUCE [37].

Вони орієнтовані на створення рішень, які кладуться в основу створення ІС, призначених для підтримки груп бізнес-процесів, які можна розглядати як один із способів описання сервісів. В їх основі компонентно-базований підхід. Архітектура цих ІС подає їх як набори компонентів, які взаємодіють для надання

системам визначеною бізнес-процесами поведінки. Сервіс у цій архітектурі є компонентом (базовий сервіс) або групою пов'язаних компонентів. У другому випадку сервіс є складним, його складовими є інші сервіси, які взаємодіють у визначений спосіб для надання ІС поведінки, якої вимагає опис складного сервісу. Кожний сервіс задається описом, який можна вважати моделлю сервісу.

Сервісно-орієнтована архітектура (СОА) програмного забезпечення ІС, яка підтримує сервіси, відокремлює сервіс, провайдера сервісу і клієнта за допомогою опису сервісу. Опис сервісу не залежить від деталей реалізації. Для клієнта він дозволяє у визначеному форматі вписати запит в термінах, побудованих на розумінні сутності сервісу. Клієнт отримує відповідь, не знаючи, як реалізований сервіс. Програми створюються шляхом збирання з готових компонентів – сервісів, які можна багаторазово використовувати як будівельні блоки. Компоненти-сервіси ІС виявляються компонентами архітектури і взаємодіють за допомогою відкритих стандартів, підтримуючи реалізацію бізнес-процесів. Спеціальна мова, синтаксис і семантика якої достатні для визначення функціональних можливостей сервісів через їх описи, дозволяє визначити передумови, постумови та поведінкову специфікацію взаємодії. Описи є основою інтеграції сервісів під час використання ІС. Клієнт може використовувати сервіси різних провайдерів, якщо їх описи відповідають запиту. Реєстр сервісів є динамічним, сервіси легко оновлювати і розширювати.

Концептуальна модель платформи PLASTIC розроблена для адаптивних послуг. Моделювання і розроблення нових сервісів базується на спільному словнику. Модель легко розширювати, адаптувати до потреб інших сервіс-орієнтованих доменів. Архітектура сервісу, підтримана W3C, уможливлює реалізацію на основі СОА. Її поширенню сприяють нові концепції – контекст, адаптація, доступні на стороні споживача ресурси, інтеграція в модель сервісу концепцій, пов'язаних з програмними компонентами. Контекст описує середовище виконання сервісу для надання і адаптації сервісу згідно вимог користувачів, критеріїв і обмежень. Опис контексту – структурована інформація про середовище з описами доступних ресурсів.

Сервіси адаптуються до контексту за налаштуваннями користувача (тип пристрою, мобільність,…) чи мережевих ресурсів (пропускна здатність мережі,…). Це дозволяє надавати послуги шляхом компромісу між потребами користувача (QoS) і параметрами поточного контексту (доступні ресурси).

Контекст забезпечує ефективне управління сервісами, їх адаптацію до потреб, оскільки враховує як функціональні характеристики (сигнатуру та поведінку), так і вимоги QoS (надійність, час затримки,…). Споживачам гарантується визначений рівень QoS в умовах неоднорідності наявних ресурсів хмарних систем. Адаптація сервісу до різних контекстів уможливлює оптимізацію ресурсів провайдера для підтримки системи сервісів.

Обчислювальне середовище управляє ЖЦ компонентів, розгортаючи, оновлюючи або видаляючи їх, не перериваючи працездатності. На рівні сервісів представлені описи сервісів та їх зв'язки, позиція, відображення між сервісами та компонентами, що реалізують їх. Розробники компонентів прив’язуються до сервісів лише на основі описів сервісів. Вибір конкретного компонента для створення екземпляра сервісу виконується з урахуванням потреб користувача і середовища/контексту доступу до сервісу.

Компонентна модель і ланцюги інструментів платформи SPICE підтримують автоматичне створення компонентів для сервісів. Компоненти поділяються на базові і комплексні – композиції базових і інших комплексних компонентів. Кожен сервіс підтримується компонентом, визначеним як фрагмент мережі базових і комплексних компонентів платформи. Можливості компонентів визначаються описами високого рівня. Компоненти та їх з’єднання не повністю відомі під час проєктування, їх вибір здійснюється автоматично відповідно до умов середовища і описів сервісів.

Платформа SPICE підтримує створення базових та інтелектуальних компонентів, додаткових сервісів. Технології автоматизації робіт із розроблення і реалізації сервісів на основі їх описів графічними мовами, емуляторів, засоби тестування суттєво підвищують ефективність створення сервісів. Досвідчені архітектори і дивелопери використовують спеціальний інструмент розроблення

із застосуванням мови SPATEL описання сервісів високого рівня, що дозволяє реалізувати підхід модельно-орієнтованої архітектури для перетворення ідеї сервісу через незалежне від платформи представлення у надаваний сервіс. Кінцеві користувачі використовують інструменти студії кінцевого користувача, які дозволяють створювати змішані сервіси, пристосовуючи існуючі сервіси до своїх потреб. Позбавлені можливості створювати довільні складені сервіси, вони з допомогою набору шаблонів можуть комбінувати і параметризувати сервіси.

У платформі OPUCE користувачі визначають нові сервіси і програми, що їх підтримують, використовуючи побудовані на відкритих службах та інтерфейсах мови, моделі та інструменти. Це дозволяє швидко створювати та розгортати нові сервіси. Користувачі, вимоги до підготовки яких помірні, створюють інноваційні привабливі сервіси, автономно керують їх ЖЦ.

Платформа підтримує можливість обміну сервісами, що дозволяє здешевити розробки і підвищити їх якість. Це вимагає від провайдерів ІКП визначити ресурси, які можуть бути надані лише в базовій мережі, а потім абстрагувати та пропонувати їх через визначені інтерфейси.

Побудована на основі СОА платформа дозволяє відкрити мережеві ресурси, використовувати веб-сервіси як проміжне ПЗ, що дозволяє третім сторонам отримувати доступ до ресурсів і контролювати їх. Користувачі створюють, керують і використовують власні сервіси, оркеструючи базові сервіси як функціональні одиниці, що розгортаються провайдером або третіми сторонами на платформі OPUCE і доступні користувачам через інтерфейси.

Основні елементи платформи: середовище створення і виконання сервісів (базові і комплексні сервіси виконуються відповідно до заданої користувачами під час створення сервісу логіки); інструменти налаштування компонентів до вимог конкретного надавача сервісів згідно їх описів і контексту, управління даними користувача, передплатами і ЖЦ сервісів.

Автоматизацію і адаптивність управління сервісами, їх створенням і наданням визначають описи сервісів. До попередніх платформ вони додають нові аспекти: функціональний (логіка, інтерфейс, семантика сервісу);

нефункціональний (SLA, якість обслуговування та ін.); керування (розклад ЖЦ сервісу, аспекти розгортання).

Зручні графічні інтерфейси, помічники, редактори та інші інструменти, дозволяють створювати сервіси у двох середовищах – повнофункціональному (мова описання з множиною правил композиції, багатий графічний інтерфейс користувача, розширені інструменти перевірки тощо) і спрощеному (менш вимогливий інтерфейс користувача, простіша мова описання, обмежений набір операцій, більш керований процес створення/налаштування тощо). Вони мають однакові принципи дизайну, працюють на визначенні сервісу.

Загальна концепція в рамках сервісного підходу спрямована на створення і використання для підтримки процесів проєктування, реалізації, впровадження, надання і розвитку сервісів платформних рішень [38-40]. Одним із варіантів є спеціалізована платформа Service Design Platform (SDP) [41]. Не так багато існує інструментів, які дають змогу реалізувати цілісну стратегію проєктування і надання сервісів [42], але є кілька цікавих прикладів її галузевого чи доменного втілення. Добрим прикладом є туристична галузь [43], де напрям розвитку SDP набув конкретизації на основі використання моделі Mobility as a Service (MaaS) [44]. Ще одним прикладом ефективного застосування SDP є платформи для проєктування, реалізації і надання програмних сервісів за моделлю Software as a Service (SaaS) [45]. Платформи SDP, орієнтовані на проєктування інфокомунікаційних сервісів, насамперед IPTV, VoIP, VPN, поштових сервісів, розсилки повідомлень та ін., представлені в праці [46]. Результати прогнозування ринку SDP говорять про його зростання [47]. Розвиток SDP показав, що важливою особливістю сервісного підходу, особливо у галузі інфокомунікацій, сьогодні є потреба у наданні пакетів сервісів і врахування при проєктуванні сервісів не тільки бізнесових, ресурсних, якісних, але й їх технічних, технологічних характеристик [48].

Розвиток SDP підтримують інтенсивні наукові дослідження, спрямовані на моделювання, проєктування, реалізацію, впровадження, надання і розвиток сервісів. Загальний опис моделей ІС компаній, які надають сервіси, на

бізнесовому, виробничому і технологічному рівнях можна знайти у працях [49, 50]. Сервісна перспектива галузі ІТ, її перехід на сервісний підхід систематизовано описано у праці [51]. Багато праць присвячено окремим аспектам дослідження сервісного підходу. Теоретичні основи проєктування сервісів викладені у праці [52]. У рамках концепції Design Science Research запропонований комплекс моделей і методів проєктування і реалізації сервісів [53]. Моделюванню і технологіям взаємодії клієнтів з сервісами присвячена праця [54]. Моделюванню середовища надання сервісів, визначенню ефективних параметрів застосування обладнання для підтримки ІТ-сервісів присвячений фреймворк, запропонований у праці [12]. Мікрорівневі механізми для створення цифрових сервісів розглядаються у праці [55]. Сучасні підходи до створення ПЗ платформи проєктування сервісів описані у працях [56 – 58].

Підбиваючи підсумки аналізу підходів до реалізації підтримки ЖЦ сервісів в ІС провайдерів можна зробити такі висновки. Універсальним рішенням проблеми вбачаються платформи підтримки сервісів, причому в матеріалах ITSM і ITIL напрацьовані основи створення ІС провайдерів в умовах впровадження сервісного підходу і дотримання концепції ЖЦ сервісів [59–60]. Для реалізації платформ розроблено низку моделей і методів, які поєднують бізнесові, технічні, технологічні та інші аспекти діяльності ІТ-компаній і провайдерів, інтереси кінцевих користувачів [62]. На рівні критеріїв бізнесовий аспект зводиться найчастіше до мінімізації загальних витрат провайдера, максимізації доходу, а технічні і технологічні аспекти побудови і використання платформ залежать від специфіки галузі [63, 64]. Усім підходам до створення платформ притаманне використання ШІ [65].

## 1.3 Проблема створення теоретичних основ підтримки ЖЦ сервісів в ІС провайдерів ІКП

Результати виконаного аналізу ЖЦ сервісів ІС провайдерів ІКП і проблем його підтримки, існуючих підходів до реалізації підтримки ЖЦ сервісів

дозволяють уточнити проблему створення теоретичних основ його підтримки і напрацювати бачення рішення як платформи підтримки ЖЦ сервісів.

По-перше, цьому сприяє комплексний характер проблеми підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Складність проблеми визначає складність рішення. Видається доцільним для вирішення складної проблеми розробити інтегроване рішення для підтримки ЖЦ сервісів і їх ефективного застосування компанією здатне врахувати усі виявлені чинники впливу:

- сервіси становлять цілісну структуровану систему, надання сервісів вимагає ресурсів, якими треба управляти з урахуванням всього комплексу сервісів і їх взаємозв'язків для мінімізації витрат з дотриманням вимог;
- процеси етапів ЖЦ сервісів в ІС провайдерів ІКП взаємопов'язані, їх реалізація є складовою етапів сервісів, яку доцільно виконувати на основі описів сервісів, шаблонів, бізнес-процесів;
- характеристики і аспекти поведінки інфраструктури ІС становлять основу для виділення віртуальних і фізичних систем для підтримки процесів ЖЦ сервісів; необхідно оцінювати їх стан для врахування всіх аспектів підтримки ЖЦ сервісів, забезпечення їх проєктування, реалізації, впровадження і надання на вищому абстрактному рівні у контексті середовища, гнучкого виділення і перерозподілу ресурсів, міграції, пошуку і усунення проблем і т.п.;
- важливу підмножину чинників впливу визначає бізнес-середовище, цілі, стратегії, організація, заінтересовані сторон, в тому числі клієнти, методи праці, бізнес-процеси та результати діяльності якого необхідно враховувати при виборі системи сервісів, її розвитку в цілому і налаштуванні складових;
- накопичені дані з підтримки бізнесової діяльності надавців і користувачів сервісів згідно прийнятого групування їх процесів за етапами ЖЦ, зокрема спожиті ресурси і економічні результати вимагають відповідних метрик і процедур їх оцінювання, щоб використати їх при постановці і розв'язанні задач підтримки зазначених процесів в ІС.

Комплексний характер проблеми підтримки процесів ЖЦ сервісів в ІС провайдерів інфокомунікацій, врахування наведених чинників впливу вимагають надання рішенню відповідних можливостей, властивостей і ознак:

1) рішення має уможливлювати інтеграцію комплексу технологій та інструментів аналізу, проєктування, реалізації, надання і розвитку сервісів з метою створення і використання ефективної ІС провайдерів ІКП;

2) рішення має базуватися на ефективних теоретичних засадах підтримки процесів ЖЦ сервісів в ІС провайдерів інфокомунікацій – проектування і реалізації, надання і розвитку сервісів, виявлення і усунення проблем і суперечностей, оцінювання сервісів, рівня задоволеності клієнтів та ін.;

3) рішення має уможливлювати швидке обчислення метрик оцінювання кількісних показників нефункціональних вимог – ефективності, доступності та інших – до ІС для моніторингу стану, керування ресурсами,…;

4) рішення має забезпечувати комплексну автоматизацію процесів усіх етапів ЖЦ сервісів в ІС провайдерів інфокомунікацій та їх взаємодії;

5) рішення має будуватися на засадах неперервної реконфігурації побудови і адаптації сервісів в ІС провайдерів інфокомунікацій;

6) рішення має бути побудоване на основі комплексу математичних моделей і методів розв’язання задач оптимізації для автоматизації усіх процесів підтримки сервісів в ІС провайдерів інфокомунікацій – ініціації, побудови архітектури, реалізації, надання та ін. – з врахуванням потенціалу і стану середовища для ефективного використання ресурсів ІТ-інфраструктури для комплексної підтримки сервісів, їх компонентів та гібридної інфраструктури хмарних систем із забезпеченням нефункціональних вимог;

7) рішення має бути узгоджене з відомими стандартизованими методологіями і підходами, насамперед ITSM, ITIL, щоб забезпечити його інтеграцію з технологіями третіх сторін, нарощування, реконфігурування, розвиток, адаптацію до змін, реалізацію концепції можливості розширення функціоналу за умови мінімізації пов’язаних з цим витрат.

По друге, результати дослідження підтвердили доцільність у сьогоднішніх умовах такого комплексного рішення за моделлю платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП.

Однак дослідження показало, що створення такого рішення має врахувати виявлені недоліки існуючих рішень. Дійсно відомим інструментам і технологіям аналізу, проєктування і реалізації сервісів, їх впровадження, надання і розвитку з урахуванням ІТ-інфраструктури, її використання і управління нею був притаманний ряд серйозних недоліків.

Розглянемо зазначені недоліки і окреслимо кроки для їх усунення. Це дозволить сформувати бачення платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП, яке буде розвинене, поглиблене і реалізоване в наступних розділах. На жаль, напрацьовані рішення не враховують усіх вимог реалізації наскрізних процесів. Цей недолік не дозволяє їм скласти основу створення ефективної платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП.

Дійсно, проаналізовані дослідження спрямовані на створення інструментів для автоматизації певних процесів. На сьогодні не сформований універсальний підхід до вирішення описаної вище комплексної проблеми управління повним ЖЦ сервісів в ІС провайдерів ІКП в умовах переходу до наскрізних сервісів з врахуванням аспектів бізнесової діяльності компаній-користувачів, провайдерів та їх клієнтів.

Також відсутні праці, результати яких можна покласти в основу аналізу і оцінювання стану ІТ-інфраструктури, щоб управляти наданням сервісів з врахуванням ресурсів. Тут бракує адекватних математичних моделей і методів визначення потреб у ресурсах, управління процесами ЖЦ сервісів з урахуванням зв'язків сервісів і процесів, впливу складових ІТ-інфраструктури на якість надання сервісів, забезпечення SLA, дотримання QoS.

Зокрема не напрацьовано математичних моделей і методів для управління розподіленими ІТ-інфраструктурами в умовах впровадження сервіс-орієнтованих архітектур ІТ-компаній, що надають наскрізні сервіси для підтримки ЖЦ сервісів в ІС багатьох провайдерів ІКП.

Загалом в основу існуючих рішень не покладені комплекси моделей, які б не тільки представляли управління повним ЖЦ сервісів в ІС провайдерів ІКП як складну проблему, але й дозволили її вирішити її, пропонуючи її обґрунтовану декомпозицію для вибору оптимальних рішень.

Отже встановлено, що теоретичного апарату для вирішення комплексної проблеми підтримки процесів ЖЦ сервісів в ІС провайдерів інфокомунікацій, сформульованої вище, в умовах впровадження в галузі наскрізних процесів, глобалізації ІТ-інфраструктур на сьогодні ще не напрацьовано. Не існує цілісної концепції інтегрованого рішення підтримки процесів ЖЦ сервісів в ІС провайдерів інфокомунікацій, підкріпленого відповідним багаторівневим набором різноманітних моделей, які враховують усі описані вище концепції.

Тому виникла потреба у ефективних засобах інтеграції рішень широкої множини розробників. Виявилося, що вони створені з огляду на різні концептуальні підходи до аналізу, проєктування і управління розподіленими застосуваннями. Ці рішення часто неефективні в умовах змішаних робочих навантажень і різнорідних ІТ-компонентів, не пристосовані для накопичення і оброблення даних в умовах змінності параметрів. Більшість з них не орієнтована на автоматизацію створення і управління сервісами на основі їх описів з урахуванням наявності кількох рівнів представлення. Це не дозволяє відстежувати тенденції в сервісах, ІТ-інфраструктурі і їх окремих складових.

Брак здатності до реконфігурування в умовах змінності вимог користувачів і середовища не дозволяє цим рішенням управляти повним ЖЦ сервісів в ІС провайдерів ІКП в умовах переходу до моделі E2E і акцентування ІТ на бізнес-потреби компаній-користувачів, провайдерів та їх клієнтів.

Встановлено, що змінність процесів ЖЦ сервісів в ІС провайдерів ІКП вимагає подальших досліджень цієї проблеми. Існуючі рішення не є інтегрованими, вони розглядають платформу підтримки ЖЦ сервісів в ІС провайдерів ІКП як традиційний об’єкт аналізу, проєктування і управління, що не вимагає принципових новацій. Однак, це не відповідає дійсності. Зазначена платформа в нових умовах є принципово новим об'єктом аналізу,

проєктування і управління, вимагає ревізії наявного теоретичного матеріалу з огляду на ієрархічність і розподіленість, великі виміри, множинність метрик оцінювання, динаміку і різноманіття вимог користувачів та інші чинники.

Тому постає потреба у розробленні концепції платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП в нових умовах з урахуванням сучасних концепцій, насамперед наскрізних сервісів.

## 1.4 Аналіз задач підтримки ЖЦ сервісів в ІС провайдерів ІКП

У цьому підрозділі в рамках описаного вище проблемного простору, визначеного потребами створення технологій сервісних платформ, розглянемо завдання дослідження, пов’язані із розробленням теоретичних основ реалізації технологій сервісних платформ. Дійсно, здатність ефективно керувати різними етапами ЖЦ сервісів є фундаментальною для успіху платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП. При цьому необхідно брати до уваги всі кроки ЖЦ сервісів, мінливість середовища надання сервісів, структуру сервісів (з точки зору впливу компонентів, важливого, наприклад, для проєктування сервісів), розподіл ресурсів з урахуванням раціональних критеріїв і обмежень та інші важливі аспекти. Зазначені аспекти тісно переплітаються, сервіси і їх зв’язки структуруються і поглиблюються.

Зрозуміло, що ми розглядатимемо усі проблеми, пов’язані зі створенням сервісної платформи, насамперед проєктування і реалізації сервісів, їх моніторингу, перевірки, наприклад для автоматичного створення наборів тестів для перевірка сервісу, розгортання в ІС провайдерів ІКП та її підтримки.

У процесі аналізу важливо знайти спільні концепції, важливі для управління сервісами у всіх галузях, щоб створити узагальнену модель для підтримки композиції сервісів, яка є спільною проблемою для різних галузей. З іншого боку, треба виділити відмінності галузі інфокомунікацій, які вимагають спеціальних моделей і методів для створення сервісної платформи саме для цієї галузі. Так, функціонування мереж провайдерів ІКП описується

з упором на інфраструктуру і рішення мають відповідати вимогам масштабованості, безпеки, неоднорідності та динамічності середовища.

Отже, можемо перейти до розроблення моделей, методів і створення на їх основі платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП для скорочення термінів впровадження ІС, зменшення витрат на їх впровадження та операційну підтримку і поліпшення їх продуктивності, відмовостійкості, придатності, розширюваності та інших якісних показників.

У роботі передбачається вирішення таких завдань:

1) підбір та розроблення метрик оцінювання сервісів, ІС, їх інфраструктури, її програмних та апаратних компонентів;

2) розроблення моделей і методів оцінювання і прогнозування стану сервісів, ІС, їх інфраструктури, її програмних та апаратних компонентів;

3) розроблення моделей і методів виявлення проблем в ІС провайдерів ІКП на рівнях підтримки операцій і бізнесу;

4) розроблення моделей і методів розв’язання задач етапів ініціації і аналізу, зокрема групування сервісів у пакети;

5) розроблення моделей і методів проєктування і реалізації сервісів;

6) розроблення моделей і методів прийняття, розгортання, передачі в підтримку і підтримки сервісів в ІС провайдерів ІКП;

7) розроблення моделей і методів розподілу ресурсів, спрямованих на надання сервісів в ІС провайдерів ІКП;

8) розроблення моделей і методів визначення бажаних змін функціональності, цінових і якісних параметрів сервісів, які пропонуються ІТ-компанією для конкретних провайдерів ІКП;

9) розроблення та впровадження платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП.

Розглянемо детальніше вимоги до зазначених моделей і методів.

### 1.4.1 Постановки задач етапів ініціації і аналізу

Розпочнемо з задач групування сервісів у пакети, ефективне розв’язання яких є важливою умовою для створення платформи підтримки ЖЦ сервісів в

ІС провайдерів ІКП. Складність проблеми обумовила існування кількох поглядів на її сутність. Перший з них базується на існуванні накопичених статистичних даних щодо отриманих прибутків за надання сервісів. Це стосується як сервісів, які надає провайдерам ІТ-компанія, так і сервісів, які провайдер надає своїм клієнтам. Загалом накопичені статистичні дані враховують рішення прийняті обома сторонами у минулі часові періоди.

Реалізація першого підходу вимагає відповідних формальної моделі і методу. Перейдемо до детального описання задачі групування сервісів у пакети у формальній постановці 1. Наведемо його на прикладі групування сервісів у пакети провайдером ІКП для клієнта. Виходимо з накопичених даних про надання сервісів у попередні періоди, поданих у вигляді:

1) множини $\{S_1, \dots, S_i, \dots, S_k\}$ сервісів, з яких провайдер ІКП групував пакети для надання клієнтам у визначений попередній період співпраці

2) кількість клієнтів сервісу $S_i, i = 1, \dots, m$, де $m$ – кількість сервісів, які надавалися у визначений попередній період співпраці;

3) ціни надання кожного сервісу для клієнта, витрати провайдера, пов’язані з наданням кожного сервісу.

Задача полягає у формуванні вибірок прибутків від надання пакетів сервісів провайдером клієнтам у визначений попередній період співпраці і визначенні статистичних показників – вибіркових середніх значень та середньоквадратичних відхилень, на основі яких ОПР може сформувати пропозицію клієнту відносно відповідного пакету сервісів. Застосування статистичного методу розв’язання задачі групування сервісів у пакети у формальній постановці 1 наведене у підрозділі 3.1.

Другий підхід базується на використанні каталогу сервісів і даних укладених у визначений попередній період угод. Тут теж визначальною є вигода ІТ-компанії (провайдера ІКП), однак дані угод характеризують попередні рішення провайдерів ІКП (клієнтів провайдерів), надають можливість визначати групи пов’язаних у часі і технологічно сервісів.

Формальну постановку 2 задачі групування сервісів у пакети теж подамо на прикладі пари «провайдер ІКП – клієнт». Визначимо дані, які будуть підставою для формування пакетів сервісів:

1) множина $\{S_1, \dots, S_i, \dots, S_k\}$ сервісів з каталогу провайдера;

2) множина $\{U_1, \dots, U_j, \dots, U_N\}$ угод провайдера з клієнтами про надання сервісів (клієнт, надані сервіси, вимоги до сервісів, час надання, ціна, витрати).

Задача полягає у формуванні вибірок вигляду $X = \| x_{ij} \|$, де $x_{ij}$ = 1 (0), якщо сервіс $S_i$ надається за угодою $U_j$ ( сервіс $S_i$ не надається за угодою $U_j$) і визначенні перспективних пакетів сервісів для клієнтів з відповідними оцінками для ОПР з метою укладання взаємовигідної угоди на новий період.

Для глибшого розуміння задачі групування сервісів у пакети у другій постановці представимо вигляд матриці $X$ таблицею 1.1. Тут легко зрозуміти зв'язок сервісів $S_i$, $i$ = 1,…,$k$, представлених рядками, з угодами $U_j$, $j$ = 1,…,$N$,, яким відповідають стовпці. І тоді цей зв'язок легко задавати одиницею або нулем на перетині відповідних рядка $i$ і стовпця $j$. Використання для розв'язання сформульованої задачі відомого підходу до групування сервісів у пакети на основі алгоритмів кластеризації наведене у підрозділі 3.2.

Таблиця 1.1 – Вхідні дані задачі групування сервісів у пакети у формальній постановці 2

| | 1 | $j$ | $K$ |
|---|---|---|---|
| 1 | 1 | ... | 1 |
| $i$ | ... | ... | ... |
| $N$ | 1 | ... | 0 |

Якщо перших два підходи переважно спрямовані на накопичений досвід надання пакетів сервісів, то третій уможливлює пошук найвигідніших варіантів групування сервісів у пакети з врахуванням усіх чинників впливу. Тут аналізуються усі можливі варіанти пакетів сервісів, у тому числі надавані у минулі періоди. Цей підхід передбачає застосування для розв'язання задачі

групування сервісів у пакети методів оптимізації. Крім того, його новаційна сутність дозволяє врахувати широке коло бізнесових, технічних і технологічних аспектів цієї задачі. З точки зору бізнес-інтересів він уможливлює прийняття рішень на основі концепції взаємної вигоди. З точки зору моделей і методів розв'язання цієї задачі мова йде про пошук компромісних рішень. Технологічні та технічні аспекти дозволять врахувати ресурсні обмеження, гарантування визначеного користувачем рівня SLA, QoS, залежність надання одних сервісів від інших та врахувати інші чинники.

Формальну постановку З задачі групування сервісів у пакети подамо на прикладі пари «ІТ-компанія – провайдер ІКП». Виходячи з необхідності врахування усіх аспектів, притаманних третьому підходу, визначимо дані, які будуть підставою для групування сервісів у пакети:

- множина сервісів $S = \{S_1, \dots, S_i, \dots, S_k\}$, надаваних ІТ-компанією множині $P = \{P_1, \dots, P_j, \dots, P_m\}$ провайдерів ІКП у складі пакетів, де $k$ – загальна кількість сервісів, $m$ – загальна кількість провайдерів;
- множини взаємозалежних сервісів $R_i = \{S_1^i, S_2^i, \dots, S_{n_i}^i\}, i = 1, \dots, k$, де кожен сервіс $S_g^i \in S, g = 1, \dots, n_i,$ є таким, що його надання можливе тільки за наявності всіх сервісів з відповідної підмножини $R_i$;
- матриця преференційних цін $D = \| d_{ij} \|$, де $d_{ij}$ – базова ціна ІТ-компанії за надання сервісу $S_i$ провайдеру $P_j, i = 1, \dots, k; j = 1, \dots, m$;
- матриця доходів провайдерів $P = \| p_{ij} \|$, де $p_{ij}$ – дохід провайдера $P_j$ від надання своїм клієнтам сервісу $S_i, i = 1, \dots, k; j = 1, \dots, m$;
- матриця знижок $R = \| r_{ij} \|$, де $r_{ij}$ – знижки до преференційної ціни $d_{ij}$ за надання сервісу $S_i$ провайдеру $P_j, r_{ij} \in [0,1), \ i = 1, \dots, k, j = 1, \dots, m$;
- $T_l$ – доступний ІТ-компанії обсяг ресурсу $l$ для надання сервісів, $l = 1, \dots, L$ ($L$ – кількість необхідних для надання сервісів типів ресурсів);
- тривимірний тензор інтенсивностей використання ресурсів $\beta = \| \beta_{ijl} \|$, де $\beta_{ijl}$ – обсяг ресурсу $l$, необхідного ІТ-компанії для надання сервісу $S_i$ провайдеру $P_j$, $i = 1, \dots, k, j = 1, \dots, m, l = 1, \dots, L$;

- матриця витрат провайдерів $B = \| b_{ij} \|$, де $b_{ij}$ – витрати провайдера $P_j$, пов'язані з наданням своїм клієнтам сервісу $S_i$, $i = 1, \dots, k, j = 1, \dots, m$;
- $G$ – кількість SLA-показників, що характеризують сервіси;
- матриця цінностей сервісів $C = \| c_{ij} \|$, де $c_{ij}$ – мінімальна відносна цінність сервісу $S_i$, $i = 1, \dots, k$, що надається провайдером $P_j, j = 1, \dots, m$;
- тривимірний тензор $A = \| a_{ijg} \|$, де $a_{ijg}$ – значення в каталозі сервісів IT-компанії SLA-показника $g$ сервісу $S_i$, що надається провайдеру $P_j$;
- матриця значень показників SLA сервісів $\alpha = \| \alpha_{ig} \|$, де $\alpha_{ig}$ – значення показника SLA $g$ сервісу $S_i$, яке визначене в умовах про надання сервісу.

*Шуканими величинами* у цій постановці задачі є:

- склад взаємовигідних пакетів сервісів для кожного провайдера, який представлений булевою матрицею $V = \| v_{ij} \|$, де

$$v_{ij} = \begin{cases} 1, \text{якщо сервіс } S_i \text{ є у пакеті провайдера } P_j, \\ 0, \text{інакше,} \end{cases} \quad i = 1, \dots, k, j = 1, \dots, m.$$

- матриця знижок $R = \| r_{ij} \|$, де $r_{ij} \in [0,1)$ – знижка до преференційної ціни $d_{ij}$ за надання сервісу $S_i$ провайдеру $P_j$, $i = 1, \dots, k, j = 1, \dots, m$.

Задача полягає у тому, щоб знайти таке групування сервісів у пакети V та знижки $R$, які забезпечують максимізацію сумарної вигоди як для IT-компанії, так і для кожного провайдера.

З урахуванням введених позначень цільові функції IT-компанії і провайдерів $P_j, j = 1, \dots, m$, задаються формулами (1.1) і (1.2) відповідно:

$$W = \sum_{j=1}^{k} \sum_{i=1}^{m} d_{ij}(1 - r_{ij}) v_{ij}, \tag{1.1}$$

$$Q_j = \sum_{i=1}^{m} (p_{ij} - d_{ij}(1 - r_{ij})) v_{ij}, j = 1, \dots, m. \tag{1.2}$$

Цільові функції (1.1) і (1.2) дозволяють оцінити вигоду, яку отримують IT-компанія та провайдери в результаті надання пакетів сервісів. Вони також уможливлюють реалізацію сучасної концепції ведення бізнесу, що полягає у

досягненні взаємної вигоди для обох сторін. У реальних умовах розв'язки, які відповідають найкращим значенням цільових функцій, обмежені. Нижче наведені формули для врахування найважливіших видів обмежень:

1) ресурсних

$$\sum_{j=1}^{k}\sum_{i=1}^{m}\beta_{ijl}v_{ij} \le T_l, l = 1, \dots, L; \tag{1.3}$$

2) обмежень на відносну цінність сервісів для провайдера

$$c_{ij}v_{ij} \le \frac{p_{ij}}{b_{ij}}, \qquad i = 1, \dots, k, j = 1, \dots, m; \tag{1.4}$$

3) обмежень на показники SLA надаваних сервісів

$$a_{ijg}v_{ij} \le \alpha_{ig}, \qquad i = 1, \dots, k, j = 1, \dots, m, g = 1, \dots, G; \tag{1.5}$$

4) обмежень залежності між сервісами, що надаються провайдерам

$$v_{ij}v_{tj} = \rho_{it}, \qquad i = 1, \dots, k, j = 1, \dots, m, t = 1, \dots, k, \tag{1.6}$$

де

$$\rho_{it} = \begin{cases} 1, \text{якщо сервісу } S_i \text{ у пакеті провайдера потрібен сервіс } S_t, \\ 0, \text{у протилежному випадку.} \end{cases} \tag{1.7}$$

Матриця $\rho = \| \rho_{it} \|$ задає всі пари взаємозалежних сервісів, які визначені складом підмножин $R_i, i = 1, \dots, k$. Як було зазначено вище, технологічні обмеження (1.6) обумовлені тим, що надання окремих сервісів може вимагати підтримки деяких їх функцій від інших сервісів.

Отже, маємо формальну постановку 3 (1.1)–(1.6) задачі групування сервісів у пакети, яка враховує особливості бізнесової діяльності ІТ-компаній і провайдерів ІКП. Вона орієнтована на досягнення власних бізнесових цілей ІТ-компанії (цільова функція (1.1)) і провайдерів (цільова функція (1.2) для кожного провайдера). Для втілення концепції взаємної вигоди введені параметри системи знижок. На разі розглянуто знижки на кожен сервіс, який надається провайдерам. Методи розв'язання наведеної задачі формування пакетів сервісів розглядаються у підрозділах 3.3 і 3.4.

### 1.4.2 Постановки задач етапів проектування і реалізації

Підтримка етапів проєктування і реалізації переважно концентрується на процесах побудови архітектур сервісів і реалізації компонентів, які забезпечують надання сервісів визначеної архітектури. Спочатку розглянемо задачу побудови архітектури сервісів.

Формальна постановка задачі побудови архітектури сервісів.

*Дано*: вимоги і навчальні вибірки.

*Необхідно знайти*: архітектуру сервісу;

### 1.4.3 Постановки задач етапів прийняття, розгортання, передачі в підтримку і підтримки сервісів

Надання провайдерам визначених на попередніх етапах сервісів вимагає розв'язання низки задач, пов'язаних з раціональним використанням ресурсів, розв'язанням проблем, які виникають в тракті надання сервісів, мінімізацією збитків від недотримання параметрів QoS, розвитком сервісів та інших. Їх розв'язання також є важливою умовою для створення платформи підтримки ЖЦ сервісів. Складність цих проблем обумовлена необхідністю врахування параметрів бізнесового, програмного і технологічного рівня архітектури ІС провайдера ІКП, створюваної за моделлю E2E.

Формальна постановка задачі розвитку сервісів.

*Дано*: вимоги і реакція соціальних мереж.

*Необхідно знайти*: нову архітектуру сервісу.

## Висновки до розділу 1

У розділі виконано перше завдання дисертаційної роботи. Виконаний аналіз сервісно-орієнтованого підходу виявив важливі для його реалізації в нових умовах аспекти. А аналіз підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП дозволив і виявити проблеми їх підтримки, насамперед появу наскрізних сервісів, яка вимагає реструктуризації існуючих платформ,

розширення їх інструментів і технологій, розвитку теоретичних основ – моделей і методів підтримки ЖЦ сервісів

Обґрунтована потреба у розробленні платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП, яка збереже переваги існуючих платформ управління ЖЦ сервісів в ІС провайдерів ІКП, дозволить інтегрувати підтримку проєктування, надання і розвитку сервісів, швидко створювати сервіси для широкого спектру потреб, сформульованих користувачами в термінах мови високого рівня, на основі комбінації базових та інших побудованих сервісів, також описаних контекстуально, впроваджувати їх, надавати і розвивати.

Сформульована комплексна проблема створення, застосування і розвитку платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП, яка в нових умовах є принципово новим об'єктом аналізу, проєктування і управління, що вимагає ревізії наявного теоретичного матеріалу з огляду на ієрархічність, інтегрованість, розподіленість, великі розміри, динаміку і різноманіття вимог користувачів, множинність метрик оцінювання, змінність технологічного, бізнесового, технічного середовищ діяльності провайдерів ІКП, інші чинники.

Обґрунтовано доцільність використання сучасних концепцій в галузях ІТ і телекомунікацій для вирішення сформульованої проблеми, зокрема комплексної автоматизації процесів усіх етапів ЖЦ сервісів в ІС провайдерів ІКП та їх взаємодії, групування процесів ЖЦ сервісів в ІС провайдерів ІКП, використання ефективних моделей і методів розширення і поліпшення функціональних можливостей, узгодженості з стандартизованими методологіями і підходами, насамперед ITSM, ITIL, пошуку і усунення вразливостей, універсалізації.

Обґрунтовано, що аналіз тенденцій розвитку сервісів і реакції соціальних мереж, забезпечення природномовної комунікації і формалізації потреб в проблемному контексті на основі аналізу природномовних текстів надасть платформі здатність допомагати користувачам вирішувати їх проблеми.

Розроблено комплекс формальних постановок задач, методи розв’язання яких складуть основу реалізації інструментів підтримки відповідних процесів ЖЦ сервісів. Запропоновано три формальні моделі задачі групування сервісів

у пакети. Перша і друга з них базуються на накопичених даних щодо надання сервісів. Третя формальна модель становить змішану задачу математичного програмування, спрямовану на пошук найвигідніших варіантів пакетів сервісів шляхом застосування методів математичного програмування і штучного інтелекту з урахуванням сучасних концепцій ведення бізнесу.

# 2 ПЛАТФОРМА ПІДТРИМКИ ЖИТТЄВОГО ЦИКЛУ СЕРВІСІВ І МОДЕЛІ І МЕТОДИ ЕТАПІВ ПРОЕКТУВАННЯ, РЕАЛІЗАЦІЇ ТА ПІДТРИМКИ СЕРВІСІВ

Виконаний у першому розділі аналіз галузі інфокомунікацій, діяльності провайдерів ІКП дозволив виявити проблеми, які вимагають вирішення. Аналіз технологій, які ІТ-компанії надають для вирішення проблем провайдерів, дозволив сформулювати потребу в універсальних інструментах створення ІС провайдерів ІКП, які враховують сучасні тенденції ведення бізнесу і забезпечують ефективну підтримку діяльності провайдерів. Обгрунтовано потребу в платформі підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП.

У цьому розділі розглядаються основи реалізації зазначеної платформи, розроблюються математичні моделі і методи, які забезпечать ефективну підтримку процесів ЖЦ сервісів, взаємодію з користувачами в процесі вирішення їх проблем, виконується експериментальне дослідження розроблених моделей і методів.

Перша група зазначених моделей і методів призначена забезпечити комунікацію користувачів з ІС провайдера ІКП в термінах розуміння їх проблем, виділення підпроблем, використання відповідних інструментів їх вирішення, інтеграції рішень і їх застосування. Тут добре себе показали LLM і RAG-системи, інтелектуальні агенти. Але врахування специфіки галузі інфокомунікацій вимагає розроблення механізмів їх тісної взаємодії.

Друга група зазначених моделей і методів спрямована на підтримку процесів етапів ЖЦ сервісів, насамперед проєктування, реалізації, надання і розвитку. Тут будуть використані нейронні мережі з глибоким навчанням для побудови архітектури сервісів. Важливо врахувати особливості структурування сервісів, широкий комплекс вимог до сервісів і характеристик провайдерів і користувачів, накопичений досвід побудови архітектур.

## **2.1** Сутність і роль платформи підтримки життєвого циклу сервісів в ІС провайдерів інфокомунікацій

Будемо дотримуватися природного порядку дослідження призначення і сутності платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП і визначимо сутність і роль зазначеної платформи у три кроки. На першому кроці обґрунтуємо використання перевірених часом ефективних концепцій побудови ІС, платформ, ІТ і їх компонентів. Але для створення платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП в нових умовах їх функціонування вони можуть виявитися недостатньо ефективними. Тому на другому кроці доцільно продумати і запропонувати нові концепції, які доповнять перевірені часом концепції, нададуть їм нових можливостей і інтегрують їх у принципово нове комплексне рішення, яке відповідає потребам розвитку галузі інфокомунікацій. Насамкінець, на третьому кроці розглянемо проблеми, пов’язані з впровадженням усіх відібраних концепцій і ефективною реалізацією зазначеної платформи.

Розпочнемо з першого кроку. Результати виконаного у першому розділі аналізу демонструють, що багато концепцій, які визначили побудову і застосування відомих платформ, збережуть свою ефективність і в нових умовах. Зокрема виявилися придатними для галузі інфокомунікацій універсальні концепції, які визначали будову і функціонування платформ в інших галузях. Насамперед, аналіз показав високу придатність для створення платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП таких перевірених на досвіді інших галузей концепцій створення платформ:

1. Інтеграційний характер платформи створює умови для успадкування можливостей платформ класу IPaaS [25]. Це фундаментально важливо для платформи підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Дійсно, понад сто процесів безумовно обумовлюють наявність великої множини компонентів платформи, які повинні взаємодіяти для надання ІС на основі платформи визначеної складної поведінки. Відповідно платформа має

забезпечувати обмін даними. Найчастіше для цього використовується API REST, але існують і інші засоби.

2. Багаторівневе описання сервісів уможливлює автоматизоване створення сервісів шляхом покрокової трансформації описів:

- зрозумілого користувачам опису сервісу на семантичному рівні в абстрактний опис сервісу, який представляє сервіс як систему складових, які взаємодіють за допомогою контрактів;
- абстрактний опис сервісу у архітектуру сервісу, яка визначає компоненти сервісу та їх взаємозв'язки.

3. Контекст сервісу пов'язує сервіс з середовищем, у якому він буде виконуватися, і представленими в ньому доступними ресурсами. Це створює умови для визначення ресурсів для компонентів архітектури сервісу, управління ресурсами з урахуванням нефункціональних вимог користувачів до сервісу, ресурсних обмежень. Крім того, це дозволяє оптимізувати використання ресурсів згідно визначених критеріїв, пристосовувати сервіс до змін середовища перерозподіляти ресурси у випадку змін у сервісі.

4. Шаблони становлять ще один спосіб спрощеного описання сервісів і формують основу для автоматизованого проектування і реалізації сервісів. Особливо ефективне використання шаблонів при проектуванні користувачами певних специфічних класів сервісів. Тут акцент робиться на підсиленні креативних можливостей кінцевих користувачів сервісів, які добре знайомі з їх практичною стороною.

5. Сервіс-орієнтована архітектура є природною для побудови платформи підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Вона використовує для взаємодії сервісів стандартні компоненти і дозволяє легко додавати нові сервіси, вносити зміни, ефективно її використовувати за рахунок процесно-орієнтованого описання її можливостей.

6. Мікросервісна архітектура у поєднанні з архітектурою керованою подіями дозволяє визначати складну поведінку ІС для підтримки процесів сервісів за рахунок взаємодії вже розроблених компонентів. При цьому

поліпшення компонентів платформи можна виконувати в рамках окремих мікросервісів. Більш того, легко виконувати заміну компонентів на інші, які мають відповідні описи.

7. Залучення ресурсів третіх сторін є надзвичайно важливою концепцією, використання якої перетворює платформу підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП у продукт, створюваний потужними спільнотами. Це сприяє скороченню термінів проектування і реалізації ІС за допомогою платформи, постійній її актуалізації до змін, повній функціональності, співпраці ІТ-компаній, яким доступні сервіси третіх сторін і можливості для надання доступу до своїх сервісів іншим учасникам. Зокрема це важливу для використання керованих ІТ-послуг.

8. Концепція використання повнофункціональної і спрощеної моделей дозволяє ефективно використовувати платформу підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП, розподіляючи опрацювання проблем співпраці між відповідними модельними рівнями, що дозволяє швидше отримувати бажані результати.

Тепер можемо перейти до другого кроку і запропонувати та обґрунтувати нові обумовлені потребами розвитку сфери надання ІКП концепції, яких вимагає галузь інфокомунікацій.

Нижче наведені нові і трансформовані під впливом змін у ІТ-галузі та бізнес-середовищі перевірені концепції. Сучасні умови використання сервісного підходу суттєво вплинули на підходи до створення, впровадження і використання платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП.

1. Стрімкий розвиток методологій розроблення інформаційних систем і технологій та їх програмного забезпечення призвів до появи комплексних рішень їх проектування, реалізації, впровадження і розвитку як сервісів. Так почали надавати програмне забезпечення і інфраструктуру, сформувалася концепція надання як сервісу зазначених інструментів проектування, реалізації, впровадження і розвитку ІС, ІТ і їх програмного забезпечення за моделлю платформи як сервісу (PaaS). Результати аналізу підтверджують, що доцільною буде реалізація підтримки процесів ЖЦ сервісів у вигляді

платформи як сервісу. Прискорювачем такої трансформації є розуміння того, що традиційні уявлення про реалізацію ІС, у тому числі в галузі інфокомунікацій, як послідовність кроків в умовах мінливого середовища повинні поступитися розпаралелюванню, тіснішій взаємодії. Це створює умови для інтеграції засобів розроблення, впровадження і використання сервісів, створення інструментів моделювання і проєктування сервісів, управління ресурсами для ефективного функціонування компонентів. Крім того, накопичення вхідних даних, даних використання платформи для підтримки процесів ЖЦ формує фундамент для застосування сучасних технологій, машинного навчання, інтелектуального аналізу і моніторингу, штучного інтелекту, великих даних та інших з метою підвищення їх ефективності.

2. Виявилося, що процеси ЖЦ сервісів різних галузей при різних інфраструктурах, хоч і досить відмінні, можна комплексно автоматизувати для підвищення ефективності. Це становить ще один чинник впливу групування процесів ЖЦ сервісів на різноманітні інструменти і технології управління ними. Постала необхідність, з одного боку, розширення складу процесів етапів ЖЦ, з іншого боку, групування процесів. Додавання процесів пов’язане з розвитком ІТ-галузі. Групування процесів дозволяє відповідно структурувати платформу підтримки ЖЦ сервісів і закласти основу для їх автоматизації.

Відомим прикладом обумовленого розвитком додавання нових процесів є реалізація згаданої вище потреби у використанні зворотного зв’язку з бізнесом і даних social media, зокрема Facebook, YouTube, з метою удосконалення сервісів відповідно до очікувань клієнтів [40]. Ще одним прикладом додавання нових процесів є поява компонентів для природномовної комунікації, виявлення змісту природномовних текстів і сприйняття природного мовлення. Це викликане потребами зручного і швидкого доступу користувачів до можливостей платформи з підтримки процесів ЖЦ сервісів. Важливі чинники впливу на додавання нових процесів становлять також вимоги проходження ІТ-компаніями сертифікації, дотримання стандартів окремих країн і економічних

об’єднань, дотримання інших тенденцій відповідних галузей, насамперед ІТ-галузі і галузі інфокомунікацій.

Що стосується групування процесів ЖЦ сервісу, то цього вимагають усі методології створення ІС і їх ПЗ. Але для платформ підтримки ЖЦ сервісів в ІС провайдерів ІКП групування процесів у етапи супроводжується групуванням процесів, які належать різним етапам. Таке групування притаманне, наприклад Mobile Services Platform. Якщо прийняти сервіс як ланцюг операцій [14–16], то природним є групування процесів за аспектами часу проєктування та виконання. У першому випадку природним буде групування процесів моделювання, розроблення та упакування сервісів. У другому випадку міжетапне групування процесів розгортання, виявлення, виконання сервісів буде вимагати додаткових засобів структурування компонентів, які реалізують етапи і процеси, їх взаємодії. Як для процесів часу розроблення, так процесів часу виконання це можна природно зробити, якщо використовується SOA. Логічним видається виділення компонентів аналізу, проєктування, реалізації, впровадження, управління сервісами, сховищем, компонентами розвитку набору сервісів та інтеграції з сервісами третіх сторін з наступним розподілом процесів підтримки ЖЦ сервісів між ними.

3. Встановлено, що суттєвий вплив на розвиток інструментів і технологій підтримки процесів ЖЦ сервісів з початку 21-го століття справляє сертифікація. Вона, з одного боку допомагає ІТ-компаніям упорядкувати процеси діяльності, оцінити і ефективно використовувати свій потенціал, поліпшити фінансовий стан, зміцнити становище на ринку. З іншого боку, досвід інституцій, пов’язаних з сертифікацією, наприклад згаданого у розділі 1 CMMI [30], показав, що для оптимізації діяльності ІТ-компаній необхідні нові комплексні інструменти. Таким інструментом в умовах впровадження сервісного підходу є згадані сервісні платформи. З врахуванням досвіду сертифікації створення зазначених платформ має базуватися на згаданих вище концепціях інтеграції, групування, автоматизації процесів ЖЦ сервісів. Але автоматизація підтримки процесів в зазначених платформах має базуватися на

чіткій системі критеріїв і оцінок ІТ-компанії, її підрозділів і напрямів діяльності, процесів і груп процесів, оцінюванні рішень, показників рівня досягнення нефункціональних вимог (ефективності, продуктивності, доступності, ...) і передусім надаваних сервісів. Введені CMMI оцінки, наприклад рівні зрілості для процесів підтримки ЖЦ сервісів доцільно використовувати, але необхідно розробити цілісну систему метрик процесів.

4. Доцільно групування процесів у платформі підтримки процесів ЖЦ сервісів виконати на спільній для найбільш авторитетних інституцій основі і надати йому динаміку, тобто важливі ознаки, правила і технології встановлення послідовності реалізації процесів, їх впровадження в систему, інтеграції, надання і розвитку в умовах контролю їх впливу на ефективність діяльності ІТ-компанії. З урахуванням організаційних, технічних і технологічних аспектів процесів ЖЦ сервісів з метою сприяння користувачам у побудові, оцінюванні і виборі альтернативних рішень згадану динаміку забезпечить розбиття процесів на такі три групи, які найчастіше згадуються у матеріалах авторитетних інституцій: ініціація, групування, побудова архітектури і імплементація сервісів; доставка, впровадження і надання сервісів; оцінювання задоволеності користувачів і удосконалення сервісів. Наведена у розділі 1 карта групування процесів в ІТ-компаніях, рекомендована CMMI, ілюструє позитивний вплив зазначеної концепції, враховуючи сутність платформи, природу процесів, їх інтеграцію, обмін даними.

5. Фундаментальною для створення і використання платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП є концепція моделювання. Економічно і технологічно невиправдане без ефективної автоматизації створення і використання зазначеної платформи вимагає моделювання як сервісів, так і підтримки процесів їх ЖЦ. Ця перевірена концепція в нових умовах вимагає модифікації. Мова йде про модель моделей, оскільки залишається згадана вище потреба у моделях сервісів, а також моделях і методах реалізації інструментів підтримки процесів усіх етапів ЖЦ сервісів. Але з урахуванням сутності сервісів і процесів їх підтримки важливо

побудувати ефективні моделі для інтеграції процесів ЖЦ сервісів, побудови композитних сервісів, оцінювання і моніторингу, пошуку і усуненню проблем і вразливостей, оцінювання задоволеності користувачів сервісів та їх потреб та інші моделі, придатні у сьогоднішніх умовах. Крім того, необхідні моделі для перетворення платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП у ефективний інструмент вирішення проблем працівників ІТ-компаній і провайдерів. А це сьогодні передбачає природномовну комунікацію, розуміння проблем користувачів і допомогу у їх вирішенні.

6. Необхідно визначати систему сервісів компанії, виходячи із особливостей бізнесу, його потреб в умовах змін бізнес-середовища, використовуючи повною мірою набутий досвід і знання, накопичені платформою дані бізнесової діяльності компанії, дані соціальних медіа.

7. Важливо використовувати гнучкі механізми автоматизації потреб користувачів у явно описані структури сервісів, які можна використовувати для управління сервісами на інших етапах ЖЦ. Однією з основних моделей, яка використовується як основа підходу до трансформації потреб користувача у описи сервісів [66], є описи бізнес-процесів мовами типу ArchiMate [67].

Тепер перейдемо до третього кроку і проаналізуємо проблеми, пов’язані з впровадженням усіх відібраних концепцій і ефективною реалізацією зазначеної платформи. Зробимо це в умовах створення платформи підтримки ЖЦ сервісів, орієнтованої на використання в ІС провайдерів ІКП, яка надається як сервіс E2E, тобто орієнтуючись на наскрізні процеси [32].

Реалізація сервісного підходу за моделлю E2E базується на наведених вище перевірені часом концепціях, доповнених новими концепціями, обумовленими станом галузі інфокомунікацій сьогодні. Сутність підходу легко зрозуміти з огляду на його призначення, основу реалізації і філософії ведення бізнесу. Призначення полягає в тому, що ІТ-компанія проєктує, реалізує, впроваджує і підтримує ІС провайдера ІКП. При цьому питання інфраструктури (спільна чи окрема для кожного провайдера), програмного забезпечення (одна копія для всіх чи своя копія для кожного провайдера),

обслуговування вирішуються залежно від вимог провайдерів. процедур – для підтримки їх бізнес-процесів. Основу реалізації складають інструменти автоматизації, побудовані на моделях і методах сервісів, процесів, їх взаємодії за умови забезпечення комплексного рішення проблем підтримки процесів ЖЦ сервісів кожного провайдера. Філософія підходу полягає у концентрації на вирішенні проблем, зменшенні і конкретизації проміжних рівнів прийняття рішень і оптимізації діяльності на основі чітко визначених критеріїв, насамперед продуктивності і ефективності бізнесу. Велика увага приділяється ефективному використанню ресурсів, усуванню проблем протидії збоям.

Уточнити сутність підходу E2E допоможе уявлення про наскрізні процеси як структуровані комплекси робіт з надання сервісів. Ідентифікація, визначення, реалізація та управління цими роботами важливе для розуміння ІТ-компанії і провайдерів ІКП з точки зору їх бізнесових цілей, стратегії, організації, методів праці, оптимізації прийняття рішень щодо підтримки процесів ЖЦ сервісів з урахуванням цілей, критеріїв, стратегії, ресурсів в умовах оцінювання і розвитку надаваних сервісів, розширення можливостей. Ключовим аспектом підходу E2E варто визначити автоматизацію процесів ЖЦ сервісів, спрямовану на забезпечення комплексної ефективності бізнесу за рахунок врахування операційної ефективності, уникнення зайвих кроків, мінімізації витрат.

Рішення для надання наскрізних сервісів має низку важливих особливостей, які необхідно реалізувати в рамках створюваної платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП:

- комплексний характер рішення – орієнтація на досягнення бізнесових цілей, опрацювання повного ланцюга для забезпечення очікуваного бізнесом результату, задоволення усіх функціональних і нефункціональних вимог, підтримка усіх процесів ЖЦ сервісів, підтримка діяльності бізнес-підрозділів, функціонування в умовах змінності вимог, обслуговування інфраструктури;

- підтримка усіх процесів створення нового сервісу – від ініціації, аналізу і планування до проектування і реалізації, і потім до надання клієнтам;
- спрямованість на оптимізацію продуктивності і ефективності бізнесу, пряма, без посередників, співпраця з клієнтами, орієнтація на комплексні сервіси.

Тепер сконцентруємося на проблемах впровадження моделі E2E і підходах до їх вирішення. Проблемами впровадження наскрізних процесів є:

1) брак комплексного рішення, яке охоплює усі процеси ЖЦ сервісів, базується на прийнятих стандартах описання сервісів, бізнес-процесів їх підтримки, надає повний спектр інструментів і технологій для проектування, реалізації, надання і розвитку сервісів, забезпечуючи моніторинг і контроль стану процесів;

2) вибір процесу E2E для ІТ-компанії і провайдерів, який найкращим чином відповідає вимогам і характеристикам ІТ-компанії і провайдерів, орієнтований на оптимізацію їх діяльності, раціональне використання ресурсів, впровадження сучасних концепцій ведення бізнесу;

3) брак пов’язаних структурування наскрізних процесів і структурування інструментів підтримки процесів ЖЦ сервісів;

4) врахування впливу соціальних аспектів ІТ-середовища, основними елементами якого є ІТ-компанії, провайдери ІКП, інституції, професійні об’єднання, спільноти користувачів.

Тому потрібне рішення для управління етапами ЖЦ сервісів провайдерів ІКП в умовах реалізації підходу E2E. Тут комплексним рішенням усіх проблем, пов’язаних із впровадженням підходу E2E видається платформа підтримки ЖЦ сервісів в ІС провайдерів ІКП, розроблена для підтримки ефективного проектування, реалізації і надання сервісів, їх моніторингу, удосконалення, гарантування функціональних і нефункціональних вимог, скорочення часу на розробку і впровадження нових процесів з можливістю взаємного використання сервісів третіх сторін.

Тепер розглянемо особливості вирішення описаних вище проблем. Першу проблему допоможуть вирішити відібрані вище перевірені і додані нові концепції створення платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП. Комплексний характер методології підтримки процесів ЖЦ сервісів з урахуванням накопичених даних дозволять взяти до уваги всі вимоги, що дозволить забезпечити комплексну ефективність, пришвидшити впровадження, мінімізувати використання ресурсів. Використання концепцій багаторівневого описання сервісів, їх контексту, шаблонів, різнорівневих моделей, мов опису бізнес-процесів і сценаріїв роботи E2E доповнюють комплексну методологію, ефективно забезпечують правильну підтримку процесів, охоплюють процеси усіх етапів ЖЦ сервісів, уможливлюють використання сервісів третіх сторін, якщо вони відповідають описам. Важливим чинником є можливість пов’язати кожен інструмент, компонент з відповідним процесом, підпроцесом чи операцією в описах бізнес-процесів чи сценаріях і підтримувати версійність інструментів.

У плані вирішення другої проблеми необхідно врахувати, що прийняті рішення мають зворотній вплив на діяльність і послуги ІТ-компаній і провайдерів ІКП. Так взятий за основу створення і використання платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП підхід Evergreen [33] гарантує відповідність сервісів, їх системи і платформи в цілому змінним умовам ведення діяльності ІТ-компанією, провайдерами ІКП. Дійсно, прийнята концепція моделювання дозволяє за наявності змін оновлювати процеси проектування, реалізації, надання та розвитку сервісів. Цьому сприятимуть ефективні моделі і методи аналізу тенденцій галузі, технологій надання і розвитку сервісів, відгуків клієнтів у соціальних мережах. Також ефективному вирішенню цієї проблеми сприятимуть обґрунтовані вище концепції використання СОА і мікросервісної архітектури, врахування функціональних і нефункціональних вимог, пошуку і усуненню проблем і вразливостей та інших.

Вирішення третьої проблеми гарантується як прийнятими концепціями, покладеними в основу створення і використання платформи підтримки ЖЦ

сервісів в ІС провайдерів ІКП, так і особливостями самого підходу E2E. Останній на основі наскрізних процесів для провайдерів ІКП дозволяє як створити окрему систему на своїй інфраструктурі, так і поділити систему на декілька віртуальних систем E2E на єдиній інфраструктурі, залежно від вимог провайдерів. Відповідно покладені в основу реалізації платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП концепції дозволяють реалізувати кожний із наведених варіантів поділу наскрізних процесів.

Для вирішення четвертої проблеми в основу створення і використання платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП покладені концепції трьох сфер. Концепції моделювання, багаторівневого описання сервісів, контексту і шаблону дозволяють виконувати заміну компонентів на нові на основі їх описів. Для ефективної підтримки процесів ЖЦ сервісів в сучасних умовах змінності бізнес-середовища додані відповідні компоненти врахування впливу соціальних аспектів ІТ-середовища, насамперед збору і аналізу опіній користувачів сервісів у соціальних мережах. Третій аспект пов’язаний з можливістю використання в рамках платформи компонентів третіх сторін, у тому числі інших ІТ-компанії, успадкованих продуктів ІС провайдерів ІКП, інституцій, професійних об’єднань.

Підсумовуючи, варто перелічити інші важливі характеристики запропонованого рішення, які розширюють його можливості: розвиток і поліпшення конкурентоспроможності сервісів шляхом врахування нових функціональних і нефункціональних вимог; удосконалення і розвитку сервісів на основі інтелектуального аналізу і моніторингу, прогнозування та моделювання; поліпшений захист за рахунок доступу до найновіших патчів безпеки; пошук вразливостей і підтримка концепції видань CI/CD; використання технологій доставляння, управління ресурсами і асинхронної взаємодії компонентів; розгортання на основних контейнерних платформах.

## 2.2 Інтелектуальний асистент платформи підтримки життєвого циклу сервісів

Запропонована у попередньому підрозділі концепція платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП надає інструменти підтримки усіх загальноприйнятих етапів ЖЦ сервісів, при цьому опираючись на моделі і методи, з одного боку для окремих процесів і груп взаємопов'язаних процесів, з другого боку, для інтегральної підтримки всієї діяльності провайдера, орієнтованої на досягнення цілей провайдера [68].

Беручи до уваги розглянуті вище тенденції розвитку галузі, можна з певністю стверджувати, що сервісний підхід в ІС провайдерів ІКП може отримати потужний імпульс для подальшого розвитку. Умовою цього є розроблення інтелектуального асистента для служб ІТ-компанії і провайдерів, здатного підтримувати усі аспекти діяльності провайдера, оцінити їх вплив на загальні показники діяльності, підказати рішення, контролювати ситуацію, пропонувати нові ідеї, спрямовані на поліпшення бізнесової діяльності.

Одним з найперспективніших підходів до реалізації асистента видається створення великої мовної моделі – Large Language Model (LLM), яка підтримує ЖЦ сервісів, орієнтуючи його на досягнення цілей провайдера і застосовуючи для вирішення окремих складових комплексної проблеми управління ефективні математичні моделі і методи. Так, оскільки одним найважливіших є етап проєктування сервісів, можна говорити про використання для цього етапу нейромережі. Для групування сервісів у пакети можна використовувати формальні моделі і математичні методи, запропоновані у працях [69-73].

Вбачається, що використання LLM може стати наріжним каменем цілісної методології підтримки сервісів в рамках їх ЖЦ сервісів, орієнтуючи запропоновану платформу на комплексну підтримку усіх процесів ЖЦ сервісів, забезпечуючи досягнення цілей провайдерів. Постає важлива наукова проблема. Розробленню LLM як інтелектуального асистента і формальних моделей та методів процесів ЖЦ сервісів, здатних скласти теоретичну основу

реалізації платформи підтримки ЖЦ сервісів в зазначених умовах присвячений цей підрозділ. Не забуваючи розвивати можливості LLM у розумінні і генеруванні природномовних текстів, зосередимося на наданні їм здатності до міркувань, вирішення проблем, прийняття рішень. Ключовими проблемами є вибір відповідної базової LLM, інтеграція ефективних методів навчання і стратегій висновків, схем прийняття рішень в процесі вирішення проблем.

Розпочнемо з аналізу сутності інтелектуальної підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Сучасні LLM здебільшого орієнтовані на розуміння і генерування природномовних текстів. Цей аспект є важливим і для використання LLM в рамках цілісної методології підтримки ЖЦ сервісів. Дійсно, тут аспект природномовної комунікації дуже важливий, оскільки вбачається, що LLM і відповідні моделі і методи будуть орієнтовані на підкріплення можливостей добре навчених фахівців – бізнес-аналітиків, власників продуктів, архітекторів сервісів і застосувань та інших – ефективно виконувати свої обов’язки. LLM досягли значного прогресу у цьому плані, щоб спілкуватися з зазначеними фахівцями, розуміти їх проблеми. Але щоб забезпечити ІС провайдера здатність надавати клієнтам ІКП швидко, з визначеним рівнем якості за ціною, вигідною провайдеру і клієнту, ефективно використовуючи ресурси, лише комунікативних навичок LLM недостатньо. Для вирішення складних проблем підтримки ЖЦ сервісів необхідні ще навички міркування, прийняття рішень. Тоді LLM, використовуючи концепцію намірів, спочатку зрозуміє проблему фахівця, а потім допоможе її вирішити або спрямувати його до відповідного інструменту вирішення проблем відповідного класу (це може бути LLM, нейронна мережа, мультиагентна система). Отже, використання LLM в рамках цілісної методології підтримки ЖЦ сервісів передбачає структурованість LLM, здатність до природномовного спілкування, застосування відомих схем і підходів до вирішення проблем шляхом їх декомпозиції, стратегій логічного висновку і оркестрації агентів.

Проблему інтелектуальної підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП ускладнює низка чинників, насамперед:

1) велика кількість взаємопов’язаних процесів, згрупованих відповідно до етапів ЖЦ сервісів;

2) комплексна підтримка процесів операційної і бізнесової діяльності;

3) багаторівневість представлення ІС з виділенням рівнів бізнес-процесів, програмних компонентів і технологій, на кожному з яких необхідно взяти до уваги велику кількість параметрів.

Ефективне врахування зазначених чинників вимагає відповідних рішень. Сформувалася потреба у новому міждисциплінарному підході до підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Такий підхід має інтегрувати всі сучасні концепції, насамперед ШІ, менеджменту.

Новий міждисциплінарний підхід склався в рамках нового наукового напряму, відомого як Service Science [74]. За короткий період часу закладено теоретичні основи аналізу, проєктування, реалізації і функціонування складних сервісних систем та розробленням відповідних мов описання і фреймворків, що уможливлюють підтримку сервісів і їх систем на всіх етапах ЖЦ.

Спільні для всіх галузей бізнесу уявлення про доставку сервісів з оптимальними показниками дієвості, ефективності і стійкості були узагальнені у вигляді комплексних методологій створення, підтримки і розвитку сервісних систем та відповідних інструментів реалізації, побудованих на основі формальних моделей і математичних методів.

Лейтмотивом Service Science є зміна концепції господарської діяльності компаній, підприємств, організацій. Раніше компоненти ІС підприємств структурувалися відповідно до концепції виробництва товарів і продуктів: бізнес-рівень (ERP, SCM, CRM, PLC, бухоблік, фінанси, кадри,…); рівень виробництва (MES, SCADA,…); процесний рівень (керування технологічними процесами).

Сьогодні бізнес орієнтується на концепцію сервісу – проблеми бізнес-діяльності розв’язуються, виходячи з переоцінки ролі сервісів. Якщо раніше центром уваги було виробництво, то сьогодні наголос робиться на послугах, операціях обслуговування, наріжним каменем діяльності є концентрація на проблемах користувачів, нових способах, методах і прийомах їх розв’язання.

У праці [75] описані особливості впровадження сервісного підходу з точки зору представлених на цьому ринку ІТ-компаній. Тут зазначений підхід втілюється в рамках створення бізнес-простору, основними елементами якого є власне сервіси, компанії-надавці і клієнти, їх потреби і цілі, ресурси, критерії їх використання. Динаміку простору забезпечує глобально керований розподіл ресурсів з метою надання клієнтам сервісів відповідно до їх потреб і цілей за критерієм взаємної вигоди компаній-надавців і клієнтів. Це стимулює модифікацію сервісів, появу нових сервісів. При такому підході беруться до уваги інтереси клієнта і вплив сервісного підходу зростає. Але одночасно постала потреба у формальних моделях, методах і технологіях підтримки усіх процесів ЖЦ сервісів з врахуванням сучасних концепцій розвитку бізнесу.

Отже, зміна концепції стала передумовою розвитку науки про сервіси. Але кожна галузь, якій притаманне надання сервісів, має свої особливості в організації і наданні сервісів. Передусім це притаманне високотехнологічним галузям, до яких належить галузь інфокомунікацій. Впровадження нового підходу до підтримки ЖЦ сервісів у ІС провайдерів ІКП вимагає врахування особливостей надання сервісів у цій галузі. Це суттєво ускладнює проблеми, які постають перед провайдерами ІКП та ІТ-компаніями, які надають засоби для вирішення проблем провайдерів [76]. Необхідно врахувати всі чинники.

По-перше, це згадані вище властивості, притаманні сервісному підходу:

1. Необхідно розробити інструменти для набору взаємопов’язаних процесів, згрупованих відповідно до традиційних етапів ЖЦ сервісів: бізнес-аналіз, управління доступністю, управління каталогом послуг, управління потужностями та ефективністю; управління змінами, управління релізами, перевірка та тестування послуг, управління розгортанням; проєктування послуг; управління безперервністю послуг; управління рівнем послуг; управління постачальниками; моніторинг та управління подіями: служба підтримки; управління запитами на обслуговування; управління інцидентами; управління проблемами; управління знаннями; управління конфігурацією ІТ-активів та послуг.

Відповідно необхідно розробити моделі і методи для реалізації зазначених процесів і моделі їх інтеграції.

2. При розробленні інструментів підтримки процесів необхідно врахувати традиційний поділ складових у рамках ІС на рівні бізнесовий і операційний. Тут вимагається розробити інструменти, які забезпечують комплексну підтримку процесів операційної і бізнесової діяльності.

3. Розроблення інструментів підтримки процесів вимагає врахування ще одного поділу ІС на рівні з виділенням рівня бізнес-процесів, програмних компонентів і технологій. Тут також є потреба у відповідних моделях і методах підтримки процесів в умовах багаторівневих представлень, при тому що на кожному з цих рівнів необхідно врахувати велику кількість параметрів .

По-друге, необхідно взяти до уваги особливості розроблення і надання сервісів, характерні для галузі інфокомунікацій:

4. Це галузь високотехнологічна. Саме тут на основі сервісного підходу користувачам почали надавати обчислювальні, комунікаційні та інші ІТ-ресурси за моделями IaaS, SaaS, PaaS та ін. В умовах віртуалізації, консолідації ресурсів в центрах оброблення даних, розвитку хмарних обчислень, СОА, реалізації компонентно-базованого підходу до створення ІС зародилася ідея глобалізації. Вона втілилася у корпоративних, національних і глобальних ІТ-інфраструктурах. Цим поняттям почали оперувати для описання організованих сукупностей взаємопов’язаних мереж, ІТ, інформаційних ресурсів і обладнання кінцевих користувачів та їх оточення, як організованої сукупності застосунків компаній-розробників, користувачів та інформаційних послуг, взаємодія яких забезпечує підтримку процесів збору, оброблення і збереження інформації користувачів. Сервісний підхід в галузі інфокомунікацій не можна розглядати без врахування інфраструктури.

5. Для галузі інфокомунікацій характерною є тенденція зростання залежності показників бізнесової діяльності провайдерів від ефективності діяльності їх ІТ-підрозділів. Бізнес чекає від останніх не просто нових технологій, йому потрібні обґрунтовані рішення для менеджерів вищої і

середньої ланок, підтримка бізнес-процесів основної діяльності. В умовах створення ІС провайдерів на основі моделі E2E відстежувати цю тенденцію мають ІТ-компанії. Відповідно зростають вимоги до надійності і ефективності підтримуваних ними ІС провайдерів. Невчасне надання послуг, як і надання неякісних послуг призводить до втрат у всьому ланцюжку «клієнт – провайдер – ІТ-компанія». Більш того, бізнес очікує, що жорсткіші вимоги бізнес-процесів до точності і своєчасності інформації будуть забезпечені більш дешевим способом. В цих умовах ІТ-компанії повинні постійно удосконалювати свою діяльність, постійно змінюватися. Додатковий донедавна домен змін у їх діяльності набуває значення визначального.

6. Характерною ознакою сучасного стану галузі інфокомунікацій є орієнтований на потреби користувачів розвиток. В умовах жорсткої конкуренції і зростання вимог замовників необхідно забезпечити певний рівень рентабельності. Сьогодні веденню бізнесу в галузі інфокомунікацій притаманні погляди на сервіс як предмет, навколо надання якого організується діяльність, важливий для клієнтів результат діяльності і спосіб поліпшення економічних критеріїв діяльності за рахунок вирішення проблем інформатизації. Розроблюються відповідні моделі, методи і технології [77, 78].

7. Щоб пристосуватися до нових умов ІТ-компанії і ІТ-служби провайдера повинні вирішити комплекс проблем, викликаних потребою: визначення змін, спрямованих на розвиток; проєктування сервісів; побудови надійної і ефективної ІТ-інфраструктури у цілому; реалізації ефективного ПЗ підтримки процесів ЖЦ сервісів; інтеграції і взаємодії ПЗ різних виробників; ефективного управління та технічної підтримки ІТ-інфраструктури та ін.

Ці процеси суттєво впливають на діяльність ІТ-компаній, які займаються розробленням, впровадженням і підтримкою технологій і інструментів для провайдерів ІКП. Тут сформувалася і системно впроваджується концепція надання сервісів за моделлю E2E. Універсальним рішенням тут видається платформа підтримки ЖЦ сервісів, описана у праці [68]. Але життя вимагає нових рішень в рамках визначених трендів. Зоопарк інструментів, що

підтримують окремі процеси ЖЦ сервісів треба впорядкувати і створити відкрите середовище, в якому комфортно почуваються працівники, які відповідають за надання сервісів користувачам.

Саме у цій галузі підтримка ЖЦ сервісів вимагає вирішення надзвичайно складних проблем. У цій ситуації треба шукати вихід. І його підказують тренди розвитку власне галузі інфокомунікацій і прогрес у розвитку таких наукових напрямів як ІТ, ШІ. І цей вихід полягає у створенні інтелектуального асистента – помічника працівників ІТ-компанії, яка розроблює, впроваджує і підтримує ІС провайдерів ІКП, і працівників служб провайдерів ІКП. Мова йде про асистента, який все пам’ятає, розуміє користувача і його проблему і допомагає її вирішити, використовуючи інформаційні, організаційні, програмні, обчислювальні, комунікаційні, математичні засоби і ресурси ІС.

Перейдемо до аналізу публікацій, присвячених створенню описаного асистента для підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Це допоможе, з одного боку, виявити всі втілені у різних продуктах корисні ідеї, з іншого боку, удосконалити існуючі рішення, наприклад, використанням для підвищення їх ефективності нових моделей або удосконаленням моделей, які не враховують важливі чинники або вимагають зміни у нових умовах.

Розпочнемо з аналізу джерел, присвячених застосуванню LLM для комплексної підтримки процесів ЖЦ сервісів. Широко представлені рішення щодо використання LLM у різних галузях переважно для природомовної комунікації з клієнтами, виявлення їх намірів і надання сервісів на основі найпростіших, переважно автоматних, моделей [79, 80]. LLM у ролі помічника у вирішенні проблем розглядається в працях [81–84]. Вони відрізняються способами направлення LLM до точного рішення. По-перше, процес міркувань з врахуванням сутності завдань і їх зв’язків поділяється на традиційні для процесу прийняття рішень кроки, які полягають у оцінюванні поточних рішень і виборі варіантів поліпшення. По-друге, щоб надати LLM здатність до багатоетапного міркування, планування дій та звертання до зовнішніх джерел використовуються мультиагентні системи. Відомі агентні рішення, такі як

Chain-of-Thought, ReAct, Graph-of-Thought, AutoGPT, BabyAGI, Plan-and-Solve Prompting, Toolformer, Reflexion, відрізняють багатокроковість, ітеративне планування наступних дій, взаємодія з різними інструментами. По-третє, використовується генерація з доповненим пошуком знань – Retrieval-Augmented Generation (RAG), за якої отримання контексту і наступна генерація відповіді на запит користувача відбуваються у одному циклі. Семантичний пошук у базі знань потрібних рядків забезпечує LLM корисною інформацією, яка допомагають формувати відповідь. Зовнішні знання надходять у вигляді наперед індексованого CSV-вмісту. Загальновідомо, що LLM у взаємодії з RAG-системою надає правильніші відповіді на питання користувачів.

Тепер наведемо джерела, в яких розглядаються формальні моделі і математичні методи процесів ЖЦ сервісів, які асистент буде використовувати для вирішення проблем користувачів. Разом з LLM, RAG-системами, іншими моделями ШІ вони складуть теоретичну основу реалізації платформи підтримки ЖЦ сервісів, ключова роль в якій належитиме асистентові.

Загальні концепції розвитку сервісного підходу у галузі інфокомунікацій, основи орієнтації ІТ-служб на проблеми бізнес-підрозділів наведені у джерелах [1, 7–10]. Джерела, присвячені вирішенню проблем створення інструментів і формальних моделей і методів підтримки окремих процесів ЖЦ сервісів поділяються на групи: відстеження галузевих тенденцій, аналізу ринку, надання здатності до самоуправління і самоорганізації [2–6]; визначення параметрів SLA, забезпечення QoS, безпеки і довіри, управління змінами [8, 55]; підтримки етапів ЖЦ сервісів, насамперед проєктування, реалізації, надання пропонуються у працях [16, 52, 53, 56, 65]; управління ресурсами і навантаженням, моніторингу та аналізу функціонування елементів ІТ-інфраструктури з врахуванням параметрів сервісів [19, 20, 61, 78].

Вирішення описаних проблем базується на адекватних методах аналізу, оцінювання, прогнозування, генерування, вибору і обґрунтування найбільш прийнятних технологій і обладнання мереж, оптимізації параметрів.

На жаль, повною мірою весь спектр перерахованих проблем ще не вирішено. Зокрема, існує потреба у комплексному підході до проєктування, реалізації і надання сервісів і розробленні відповідних інструментів – платформ, ІТ, продуктів і компонентів. Рівною мірою належить приділити увагу моделям і методам, які мають скласти основу для створення зазначених інструментів. При цьому перспективною видається інтеграція в рамках LLM інструментів, побудованих на основі нейромереж, моделей машинного навчання, прийняття рішень і математичного програмування та інших LLM.

Тепер перейдемо до постановки проблеми створення інтелектуального асистента для підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП. Постає проблема підтримки процесів ЖЦ сервісів, якій притаманні:

1) реалізація інструментів підтримки десятків наведених вище процесів, кожний з яких вимагає специфічних моделей, методів і технологій;

2) пов’язаність усіх процесів, яка створює загальну картину діяльності провайдера у всіх аспектах – бізнесовому, організаційному, функціональному, інформаційному, програмному, технологічному;

3) підтримка процесів ЖЦ сервісів, здійснювана на основі моделі E2E з врахуванням динаміки розвитку галузі, успадкуванням трендів, концепцій і ідей, які з цим пов’язані.

Треба розробити технологію, яка: спілкується з користувачами-працівниками ІТ-компанії і провайдера, які відповідають за підтримку усіх процесів ЖЦ сервісів, природною мовою; розуміє проблему; допомагає зібрати усю потрібну інформацію; розкладає проблему на складові підпроблеми; підказує компонент підтримки ЖЦ сервісів, до якого треба звернутися для вирішення кожної підпроблеми; інтегрує розв’язки підпроблем у розв’язок поставленої проблеми.

Насамкінець опишемо загальне рішення зі створення інтелектуального асистента підтримки процесів ЖЦ сервісів. Впровадження бізнес-процесів як основи проєктування сервісів їх підтримки в ІС провайдерів ІКП, створюваних за моделлю E2E, передбачає розроблення рішення у вигляді інтелектуального

асистента, який бізнес-процеси і бізнес-вимоги користувачів перетворює у налаштування платформи таким чином, щоб підтримати сервіси компонентами і технологіями, на яких вони базуються. Для економії часу і швидкої перевірки правильності закладених ідей будемо використовувати стандартну LLM, яка буде додатково структурована і навчена. Мова йде про власне модель моделей:

1) структуровану LLM кореневого рівня, яка знає всі процеси і їх зв'язки, входи і виходи, спілкується з користувачем природною мовою, розуміє наміри користувача і підказує до послуг якої LLM першого рівня йому треба звернутися (використовуючи традиційний синхронний REST API);

2) множину LLM першого рівня, які створюються для кожної групи процесів, кожного процесу і навіть, за необхідності, підпроцесу. Кожна з них знає всі процеси групи, для якої створена, відповідний процес чи підпроцес ЖЦ сервісів, їх зв'язки, виходи і входи, спілкується природною мовою і підказує наступні кроки команді, яка підтримує ЖЦ сервісів, керуючи процесом вирішення їх проблеми.

Структурована Large Language Model кореневого рівня повинна реалізувати загальну схему діяльності компанії, спрямованої на підтримку ЖЦ сервісів. Ця LLM структурується на основі традиційних понять застосування відомих схем і підходів до вирішення проблем шляхом їх декомпозиції, стратегій логічного висновку: проблема, підпроблема, причина, наслідок, передумова, варіант рішення, вибір варіанту рішення і т.д.

Структурована Large Language Model кореневого рівня навчена розуміти призначення кожної групи процесів, кожного процесу і підпроцесу, їх входи, виходи, зв’язки, стан об’єктів контролю, управління, оцінювання, проблеми контролю, управління, оцінювання, засоби (LLM, нейромережі, програмні компоненти, тощо), які використовуються для вирішення проблем.

LLM першого рівня, створена для кожної групи процесів, кожного процесу і навіть деяких підпроцесів підтримуватиме діалог, виконуватиме роль асистента у реалізації відповідного підпроцесу і за допомогою REST API викликатиме інші LLM чи ПЗ платформи, побудоване на традиційних моделях,

методах і технологіях ТПР, дискретної математики, інших моделях ШІ (це можуть бути нейронні мережі з глибоким навчанням, математичні моделі оптимізації, моделі на основі систем правил, еволюційні алгоритми, тощо).

Однак, важливою особливістю ІС провайдерів ІКП є необхідність надання точних відповідей на специфічні запити користувачів, перед якими постають складні проблеми підтримки ЖЦ сервісів. Сучасні LLM є важливим кроком у розвитку систем ШІ, але досвід показує, коли необхідні конкретні числові розрахунки, розв'язки оптимізаційних проблем, вирішення проблем на основі певної логіки їх декомпозиції, прийняття рішень, LLM треба прищепити нові можливості. Щоб навчити LLM співпрацювати з користувачами в термінах вирішення проблем, їх треба інтегрувати з зовнішніми базами знань.

З цією метою LLM інтегрують з RAG-системами. Генерація з доповненим пошуком знань передбачає надання LLM перед формуванням відповіді адекватного фрагменту інформації з відповідно до поставленої проблеми структурованого і заповненого зовнішнього джерела на основі ефективних засобів швидкого доступу [82]. Найчастіше для цього використовуються бази даних або документів. Таким чином, LLM формує відповідь в рамках визначеної проблеми, ґрунтуючи її на фактичних даних. Це допомагає користувачам інтерпретувати відповідь в термінах поставленої проблеми і зменшує ймовірність галюцинацій моделі. Поєднання генеративних можливостей LLM з можливостями RAG-системи дозволяє інтелектуальному асистенту взаємодіяти з базою знань для надання точних, обґрунтованих і актуальних відповідей користувачам. Завдяки цьому LLM використовує і внутрішні знання, і отримує доступ до БД. Тим самим інтеграція LLM з RAG перетворюється у ключовий компонент підтримки ЖЦ сервісів, який гарантує релевантність результатів у реальному часі.

У четвертому розділі буде наведена реалізація RAG-підходу, яка базується на використанні для надання LLM здатності до розв'язання проблем за рахунок роботи з даними у форматі CSV. Ми опишемо RAG-систему, яка завантажує таблицю CSV з назвами пристроїв IBM, їх характеристиками, і

надає LLM доступ до цих табличних даних, щоб швидко генерувати релевантні відповіді на специфічні запитання користувача.

Для реалізації запропонованого рішення необхідно описати:

1) кожний елемент, вхід і вихід елемента, моделі і методи, які у ньому доцільно використовувати;

2) рольову участь працівників провайдера ІКП.

Інтелектуальний асистент на основі базових сценаріїв і шаблонів операцій для надання сервісів за моделлю E2E буде реалізувати підтримку процесів ЖЦ сервісів у взаємодії з працівниками провайдера ІКП у такій послідовності: 1) Аналіз; 2) Визначення вимог; 3) Проєктування (огляд та уточнення сценаріїв та процесів експлуатації; підготовка чорнової версії для кожного підланцюга E2E; визначення додаткових вимог до команди CSE); 4) зіставлення кожного інструменту/вимоги з конкретним сценарієм операції E2E; 5) Прийняття (виконати попереднє ORT-приймання та приймання ORT на основі сценаріїв операції E2E).

Процеси, для виконання яких у ІС провайдерів мають бути інструменти, наведені вище у підрозділі 1. Підтримка процесів ЖЦ сервісів в ІС провайдерів ІКП, створюваних за моделлю E2E, забезпечується засобами платформи підтримки сервісів, які приводяться в дію інтелектуальним асистентом.

Інтелектуальний асистент налаштовує і використовує засоби платформи підтримки сервісів, розуміючи проблеми, для вирішення яких вони використовуються, підтримуючи комунікацію з користувачами. Мова йде про програмні компоненти, які реалізують відповідні моделі і методи дискретної математики, ТПР, математичного програмування і моделі ШІ для підтримки окремих процесів, підпроцесів і операцій. Таким чином, інтелектуальний асистент становить модель моделей, яка інтегрує усі корисні компоненти ПЗ для моделювання раціональної поведінки ІТ-компанії і провайдера.

Так, проєктування і розвиток сервісів в ІС провайдерів може виконуватися у дві стадії: 1) з урахуванням функціональних вимог та/або призначення сервісу вибирається відповідний шаблон із набору рамкових

шаблонів; 2) за допомогою нейронної мережі підбираються додаткові елементи і зв’язки, які використовують накопичені знання і враховують нефункціональні вимоги. Для цього у нейромережі з врахуванням специфічних особливостей проблеми необхідно відповідним чином реалізувати процеси: 1) навчання (навчити мережу підбирати нові, більш відповідні до вимог рамкові шаблони і типові сервіси); 2) класифікації (навчена мережа підбирає рамковий шаблон і типовий сервіс на основі функціональних вимог та/або призначення сервісу і нефункціональних вимог і додає до нього додаткові елементи).

Тому важливо, щоб нейронна мережа з глибоким навчанням включала усі необхідні елементи, зв’язки, базові шаблони, типові сервіси, вимоги, а на виході видавали структуру сервісу, яка найкращим чином відповідає вимогам і призначенню. Тут треба орієнтуватися на навчання мережі використанню найбільш вдалих компонентів і зв'язків для формування необхідних архітектур, враховуючи вимоги і призначення. У нейромережах цього можна досягти шляхом виявлення в архітектурі цих сервісів, у вимогах до них та інших складових спільних елементів, взаємозв’язків, схем поведінки тощо і формування на цій основі нової архітектури сервісу.

Загалом система з інтелектуальним асистентом на верхньому рівні моделює традиційне дерево рішень, яке базується на функціональних вимогах до сервісів і визначає набір продуктів та їх компонентів, які будуть використані для надання сервісів провайдеру, а на другому рівні вибирає і застосовує нейронні мережі та традиційні моделі, методи і технології для конфігурування та визначення параметрів. На верхньому рівні реалізується логіка розв’язання проблеми підтримки діяльності, пов’язаної з організацією взаємодії процесів усіх етапів ЖЦ сервісів (проєктування, розвиток та ін.). На нижньому рівні реалізується логіка виконання окремих процесів, підпроцесів, операцій на основі визначених на верхньому рівні вхідних даних і вибраних технологій.

Загальний вигляд архітектури платформи підтримки процесів традиційних етапів ЖЦ сервісів в ІС провайдерів ІКП з урахуванням процесів, яких вимагає сертифікація CMMI, підходу Evergreen та концепцій

інтегрованості платформи і керованих сервісів з використанням інтелектуального асистента для інтеграції компонентів і комунікації з користувачами в ході розв’язання їх проблем наведений на рис. 2.1. В основу архітектури покладена концепція мультиагентної системи з виділенням агента-комунікатора і агента-оркестратора відповідного рівня, а також агентів, які як компоненти відповідають за вирішення підпроблем.

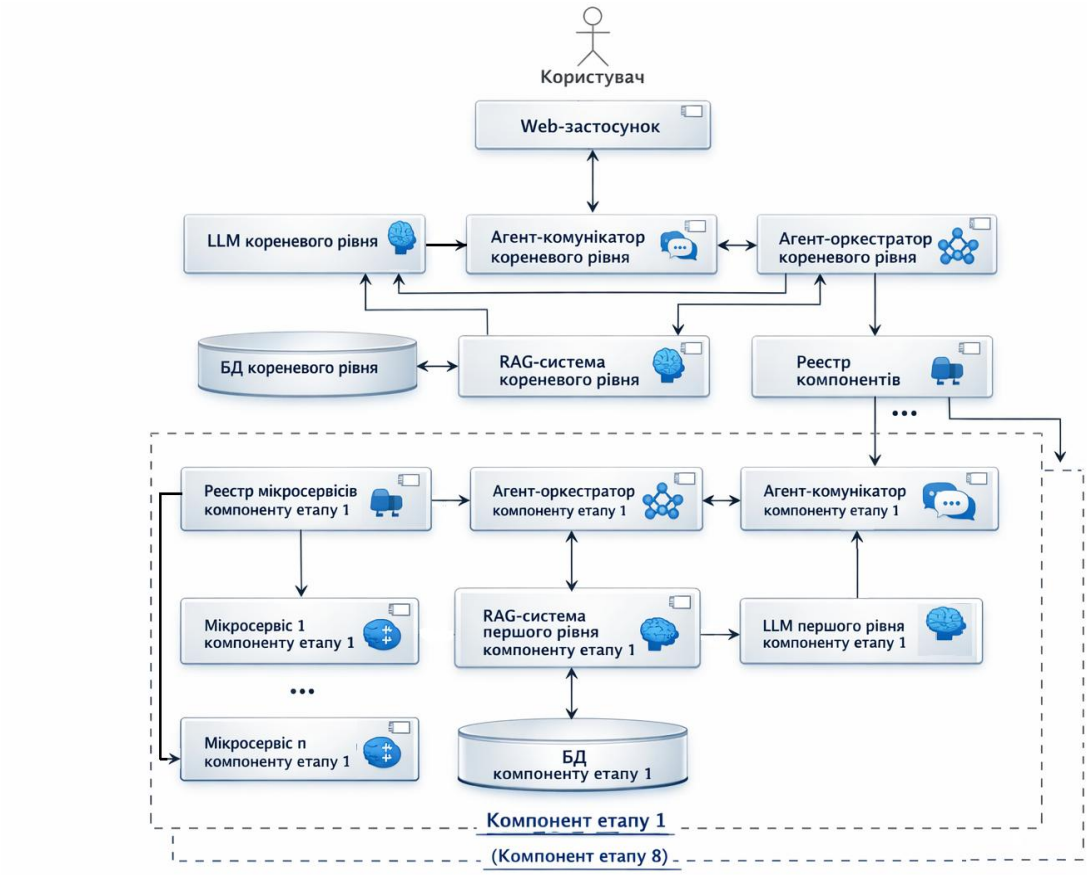


Рисунок 2.1 – Архітектура платформи підтримки процесів традиційних етапів ЖЦ сервісів

Для реалізації такої системи необхідно використовувати кілька важливих джерел даних – тип і параметри сервісу, функціональні вимоги до сервісу, особливості клієнтів (розмір, складність, домен,…), нефункціональні вимоги, тощо. Це дозволить вибрати базову архітектуру сервісу для клієнта, яка потім розширюється шляхом додавання нових елементів, зв'язків, шаблонів, типових сервісів. Тобто в системі здійснюється регулярний аналіз вже відомих сервісів, розраховуються оцінки цих сервісів, визначаються прийнятні, спільні вимоги

до сервісів. Таким чином, кожен сервіс в системі оцінюється в процесі його надання в режимі реального часу, а крім того, для кожного сервісу розраховуються статистичні оцінки ефективності та інших важливих для прийняття рішень на етапі бізнес-аналізу параметрів. Важливо мати зазначені оцінки для кожного сервісу загалом і для цього ж сервісу стосовно кожного клієнта. Тому необхідно детально описати сервіси, функціональні і нефункціональні вимоги, архітектури, структурні елементи і зв'язки, впровадити відповідну систему збору інформації, оцінювання ефективності та інших важливих для бізнесу показників для ІТ-компанії, провайдера і клієнта. Важливі також оцінки користувачів, отримувані через соціальні мережі.

Реалізація такої системи породжує низку проблем, наприклад управління ресурсами для ефективного функціонування інтелектуального асистента, платформи, нейромереж, інших технологій. Необхідно збирати архітектури і оцінки від ІТ-компаній, провайдерів і клієнтів у стандартну векторну форму.

## 2.3 Структурована велика мовна модель кореневого рівня як основа інтеграції компонентів платформи і комунікації користувачів з нею

Отже, для досягнення сформульованої мети необхідно надати LLM здатності до розв’язання проблем підтримки процесів ЖЦ сервісів. Виконаний аналіз показав, що загалом LLM надають користувачеві можливість отримати цікавий матеріал, який сприяє розв’язанню певної проблеми. Особливістю цього підходу є використання природномовних текстів, як даних, які підбираються для відповіді на очікувані питання, тобто по суті навчального корпусу як основи розв’язання проблем користувачів, так і даних, які генеруються як відповіді на питання, тобто розв’язки проблем.

Однак дослідження безпосереднього використання LLM для розв’язання проблем підтримки процесів ЖЦ сервісів, притаманних ІС провайдерів ІКП, дозволило виявити певні обмеження такого використання LLM.

По-перше, показано, що можливості LLM правильно відповідати на питання насамперед визначаються якістю навчального корпусу [79]. Тому

вважається, що творці моделі мають розуміти проблеми, які за її допомогою будуть намагатися вирішувати користувачі і підібрати досить багато відповідних наборів даних, які можна розглядати як лінії поведінки при розв'язанні проблем в певних проблемних ситуаціях. Але практика показує, що підходу до розв'язання проблем шляхом накопичення створених людиною навчальних наборів даних притаманна нестійкість. Дійсно, експерименти показали, що надійність системи, тобто здатність надавати правильні розв'язки при доданні нових навчальних матеріалів падає. Це пояснюється руйнівним впливом зростаючої складності на стабільність процесу отримання ефективних відповідей LLM – розв'язків проблем.

По-друге, встановлено, що LLM досить ефективні при генеруванні природномовних текстів як відповідей на окремі, навіть досить складні питання, але при розв'язанні комплексних проблем LLM не такі ефективні [80]. Але ж підтримка процесів ЖЦ сервісів вимагає саме розв'язання комплексних проблем. У рамках певної методології тут потрібно застосовувати знання з математики, бізнесу, телекомунікацій, дотримуючись звиклої людині здатності до міркувань, Інакше кажучи, необхідно прищепити LLM здатність міркувати, якщо ми хочемо, щоб вона перетворилася у інтелектуального асистента при розв'язанні складних проблем підтримки процесів ЖЦ сервісів. Тобто LLM мають стати чимсь більшим, ніж простим перетворювачем і генератором тексту. А це вимагає застосування технологій навчання, ефективних стратегій висновків для імітації процесу міркувань.

По-третє, при вирішенні складних проблем підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП часто використовуються напрацьовані графічні матеріали, наприклад описи бізнес-процесів графічними мовами, схеми алгоритмів, структурні, функціональні, організаційні схеми, компонентно-поєднувальні і багаторівневі шаблони архітектури систем, документація з GUI, тощо. Хоч, як говорилося вище, LLM досить швидко можна навчити ефективно обробляти текстові дані, їх використання для інтерпретації, розуміння і відповідного використання графічних елементів породжує значні

труднощі. Як наслідок, зменшується їхня ефективність при виконанні завдань, де порядок дій, вхідні і вихідні дані, інша корисна інформація подається у графічному і часто візуальному вигляді, надто коли є потреба безперебійної взаємодії. Тут використання LLM потребує додаткових рішень [81].

Тож загалом прийнято вважати, що LLM найбільш придатні для у випадках роботи зі складними неструктурованими текстовими даними, в умовах не формалізованих динамічних випадків використання та природномовної комунікації з користувачем. Якщо виникає потреба працювати зі структурованими даними, то загальноприйнятним рішенням є використання традиційних спеціалізованих інструментів.

Але при вирішенні складних проблем підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП найкращим видається гібридний підхід. Тут LLM доцільно використовувати для опрацювання великої кількості текстових даних, розроблених в умовах впровадження концепції реінжинірингу. Для всіх спеціальних випадків використання ШІ доцільно до LLM додати спеціалізовані інструменти, орієнтовані на певні формальні підходи до розв’язання проблем. Дійсно, гібридний підхід дозволить ІТ-компаніям і провайдерам ІКП ефективно використовувати інструменти аналітики і LLM. Це дозволить збільшити ефективність і зменшити споживання ресурсів.

Для ефективної реалізації гібридного підходу необхідно шукати нові концепції для ефективного використання LLM у процесі розв’язання проблем підтримки процесів ЖЦ сервісів, притаманних ІС провайдерів ІКП.

Для усунення впливу першого обмеження розроблено багато методів. Так, рішення на основі концепції самовдосконалення забезпечують підтримку безперервного циклічного процесу запитання-відповідь до досягнення потрібної точності згенерованих відповідей. Їм притаманні масштабованість і економічність завдяки зниженню вимог до об’ємів наборів даних.

Однак, виконані дослідження концепції самовдосконалення для розв’язання проблем підтримки процесів ЖЦ сервісів говорять про необхідність удосконалення її реалізації, щоб запобігти насиченню вже після

небагатьох ітерацій. Ефективним підходом тут є правильне поєднання дослідження (exploration) та експлуатації (exploitation). Щоб надати моделі здатність генерувати правильні та різноманітні відповіді пропонується реалізувати ефективну стратегію зовнішніх винагород. Наслідком використання невдалої стратегії будуть спад зростання і, як наслідок, зменшення ефективності моделі.

Загалом визнано, наприклад [80], що ефективність LLM забезпечує поєднання відповідних метрик балансу, способів їх оцінювання, методів максимізації середньої оцінки балансу шляхом динамічного налаштування конфігурацій параметрів моделі. Експерименти показали, що ці засоби запобігали сповільненню та стагнації, сприяли тому, що LLM частіше давала правильні, якісніші відповіді при розв'язання проблем підтримки процесів ЖЦ сервісів. Однак, була підтверджена потреба у засобах розв'язання проблем.

Для усунення другого обмеження використовуються математичні моделі прийняття рішень для моделювання процесу міркування. Для цього спочатку необхідно розробити засоби виділення завдань або проблем і їх розуміння, використовуючи традиційні терміни мети, входу, виходу, засобу. Потім необхідно розробити засоби декомпозиції проблеми на підпроблеми та інтеграції розв'язків підпроблем у розв'язок проблеми. Тут важливо, щоб декомпозиція і розв'язки проблеми оцінювалися та оптимізувалися у такий спосіб, щоб систематично направляти LLM до точного рішення. Напрацьовано багато рішень, наприклад [79-81], які базуються на використанні досить ефективних концепцій розширення складу даних для контролю процесу, онлайн-навчання з підкріпленням (RL), моделей, наприклад Gen & Discriminative, винагороди процесів, керованого часом пошуку для посилення здібностей міркувати з використанням багатопошукових стратегій, обчислення та масштабування під час тестування та ін.

Дослідження багатьох шляхів міркування на кожному етапі розв'язання проблеми дозволяє вибрати найкращі способи структурування проблем і формування рішень, враховуючи не лише рішення для окремих завдань, але і

проблеми в цілому. Це й формує основу для здатності LLM набути навичок логічного міркування, а також сприяє отриманню все кращих розв'язків з урахуванням класів проблем. У цьому випадку можна говорити про ефективність і надійність процесу міркування.

Винагорода дозволить LLM приймати рішення проблеми за оцінками результатів проміжних кроків, а не тільки за кінцевим результатом. Цьому сприяє структурування проблеми. На основі зворотного зв'язку у процесі міркувань LLM реалізує модель прийняття рішень на основі декомпозиції проблеми, інтеграції рішень підпроблем. Це надає мережі здатність налаштовувати процес міркувань для отримання ефективніших рішень.

Структурування процесу розв'язання проблем також дозволить закріпити за LLM здатність покроково міркувати, навчити модель не тільки масштабувати параметри, а підбирати ефективніші стратегії висновків.

Експериментально підтверджено, що структурування процесу вирішення проблеми значно підвищує ефективність і точність LLM при використанні на навчальних наборах даних у порівнянні з традиційними підходами. Доведено, що LLM можна використовувати для підтримки прийняття рішень, коли вимагаються універсальність, ефективність. Вони пристосовані до навчання, застосування сучасних моделей прийняття рішень. Цьому також сприяє управління пошуком під час тестування та впровадження LLM, можливість вибору методів винагороди, стратегій налаштування процесу прийняття рішень з врахуванням багатьох шляхів міркувань.

Для зменшення впливу третього обмеження LLM необхідно надати здатність розуміти графічні матеріали. LLM повинна вміти перетворювати графічні зображення у структуровані машино зчитувані дані, які визначають певні схеми розв'язання проблем з відповідними підпроблемами, їх зв'язками на основі вхідних і вихідних даних, певними умовами, виконавцями, термінами, обмеженнями та іншими елементами прийняття рішень. Це дозволить LLM ефективніше розуміти та взаємодіяти з різними програмними

інтерфейсами, використовувати накопичені графічні дані, поєднати опрацювання текстових і графічних даних у одній комплексній системі ШІ.

Для опрацювання графічних даних використовуємо загальноприйнятий підхід. Для кожної графічної мови маємо два компоненти: компонент виявлення, який ідентифікує елементи графічного зображення, наприклад бізнес-процеси, підпроцеси, операції, види зв’язків, тощо; компонент описів, який формує текстові описи елементів зображення, надаючи контекст щодо їхніх функцій в платформі підтримки процесів ЖЦ сервісів. Це дозволить LLM створити детальне розуміння графічного матеріалу і допомогти користувачам взаємодіяти та виконувати завдання, розв’язувати проблеми.

Ефективність опрацювання графічних даних також визначається навчальними наборами даних. Важливо навчати систему на широкому та вдосконаленому наборі даних для кожної графічної мови, з якою працює LLM. Добре підібраний набір даних підвищує точність моделі у виявленні та описанні елементів, які є важливими для взаємодії з користувачами і розв’язання їх проблем. Доцільним видається оптимізувати розміри зображень, які оброблятиме LLM, щоб скоротити затримки і збільшити продуктивність та забезпечити достатню точність. Це надасть LLM здатність точно інтерпретувати та взаємодіяти зі складними графічними зображеннями.

Це також можна розглядати як основу створення інтелектуальних агентів опрацювання графічного матеріалу, а потім і агентів, здатних автономно орієнтуватися та маніпулювати графічними інтерфейсами. Це шлях до створення більш інтуїтивно зрозумілих та потужних систем ШІ.

Дослідження у цій перспективній сфері доцільно спрямувати на пошук правильного балансу між потужністю і практичністю. Інтелектуальний асистент має використовувати ефективні інструменти для роботи, а не просто сліпо копіювати останні тенденції. Рішення мають ґрунтувалися на потребах підтримки процесів ЖЦ сервісів з використанням всіх накопичених знань.

Ключовими аспектами, які виділяють інтелектуального агента як помічника для працівників служб ІТ-компанії і провайдерів, варто прийняти:

1) структурування діяльності, пов’язаної з підтримкою ЖЦ сервісів (в термінах процесів, підпроцесів, їх зв’язності, входів і виходів, ролей);

2) розроблення і реалізація конкретного наповнення підтримки вже для кожного процесу (до традиційні кроків – збір даних, генерування варіантів, прийняття рішень, з часом будуть додані нові концептуальні кроки);

3) для кожного процесу LLM підключатиме одну чи декілька моделей для підтримки власне процесу і підтримки ролей, які відповідають за нього;

4) моделями можуть бути інші LLM, нейромережі і інші моделі ШІ, моделі математичного програмування та дослідження операцій, інші моделі (як і природний інтелект, ШІ повинен використовувати у загальній схемі розв’язання проблем складові, які опираються на перевірених математичних моделях і методах генерування варіантів, оцінювання, прийняття рішень).

Підводячи підсумки структурування LLM кореневого рівня, загалом доцільним видається пропозиція розроблення інтегрального рішення для комплексної підтримки процесів ЖЦ сервісів, побудованого на розглянутих вище і нових концепціях, спрямованих на усунення обмежень:

1. Структурованість ЖЦ сервісів (десятки пов’язаних процесів, поєднані у 10 основних етапів) обумовлює розроблення узагальненої LLM кореневого рівня, яка буде пов’язувати LLM нижніх рівнів і моделі інших типів, які створюються для окремих етапів і процесів ЖЦ сервісів. Це складе основу для співпраці між спільнотою, яка підтримує ЖЦ сервісів, та інтеграції процесів і етапів ЖЦ сервісів, розвитку можливості міркувань;

2. Кожній LLM придаємо здатність до міркувань за рахунок керованого пошуку при розв’язанні проблеми. Тут важливо зберегти здатність LLM до швидких автоматичних відповідей і надати їй здатність до логічних міркувань

3. Впровадження вдосконалених методів навчання з підкріпленням забезпечить комплексну та відкриту платформу для використання LLM для підтримки ЖЦ сервісів.

4. Впровадження комплексу підходів до структурування проблем і управління процесом висновків дозволить оптимізувати вибір підходу для розв’язання проблем, притаманних окремим етапам і процесам.

5. Використання RAG-систем для доступу до зовнішніх джерел даних.

6. Структурування LLM з метою надання здатності до розв’язання проблем є використання контексту підтримки процесів ЖЦ сервісів.

7. Створення на основі LLM та інших моделей ШІ агентів, які у взаємодії забезпечують комплексну підтримку процесів ЖЦ сервісів.

## 2.4 RAG-система як основа реалізації проблемного підходу у платформі підтримки життєвого циклу сервісів

Отже, застосування RAG-системи спрямоване на поліпшення здатності LLM генерувати відповіді на складні запитання, пов’язані з підтримкою процесів ЖЦ сервісів. Використана вперше з цією метою RAG-система інтегрувала генеративні моделі та механізми пошуку інформації [82]. Це дозволило надавати точніші і повніші відповіді на складні запитання. Оскільки практичне застосування нового підходу вимагало все кращих показників, було запропоновано кілька його удосконалень. Серед них варто виділити дослідження, які базуються на концепціях:

- використання інформації із зовнішніх джерел для структурування відповідей LLM у рамках контексту розв’язання проблем визначеного класу [86];
- застосування активного навчання і динамічного вибору найбільш релевантних документів [87];
- повторного використання вже знайдених даних та інтерактивного уточнення запитів [88];
- опрацювання джерел з різними формами представлення інформації – текстовою, графічною, табличною (мультимодальність), наприклад [89].

Аналіз сучасних рішень, в яких застосовуються RAG-системи, показав, що для ефективної підтримки ЖЦ сервісів у ІС провайдерів ІКП є потреба у реалізації RAG-системи, яка використовує переваги всіх сучасних концепцій поліпшення ефективності LLM генерувати відповіді на складні запитання, зокрема мультимодальності. Методи Retro [90] і Toolformer [91] базуються на інтеграції статичних знань та інструментах пошуку без мультимодальності.

Але існують методи, які доповнюють текстовий пошук  пошуком в зображеннях (MuRAG [92] з мультимодальною пам'яттю  для відкритого домену QA), ієрархічним пошуком в мультимодальних документах (Wiki-LLaVA [93], візуально-центричним пошуком (VisRAG [94]) з використанням VLM, відповідями на прості і складені питання у домені QA з опрацюванням тексту, графіків і рисунків (M3DocRAG [95]), паралельним візуальним каналом для підтримки багатокрокового процесу пошуку відповідей на питання (VisDoMRAG [96]. Ці системи відрізняються реалізацією, але демонструють, що розв'язання проблем підтримки ЖЦ сервісів має базуватися на:

- їх декомпозиції на підпроблеми, розв'язанні підпроблем та інтеграції розв'язків;
- використанні накопиченої у текстовій, графічній і табличній формі інформації ІТ-спільнот, у якій узагальнено досвід розв'язання проблем провайдерів;
- застосуванні реальних сценаріїв діяльності, в яких структуруються процеси (виділяються підпроцеси і операції, їх зв'язки, умови, виконавці, інструменти);
- комплексному опрацюванні інформації з багатьох джерел.

Отже, для ефективної реалізації платформи підтримки ЖЦ сервісів доцільно реалізувати мультимодальну RAG-систему на основі агентної логіки з багатокроковим логічним виведенням.

У підрозділі розглянемо побудову архітектури такої агентної мультимодальної RAG-системи, виконаємо її експериментальне дослідження та порівняльний аналіз з традиційною текстовою RAG-системою. Отримані

результати їх оцінювання за показниками точності, повноти, продуктивності та зручності використання дозволять запропонувати практичні рекомендації щодо впровадження агентних мультимодальних RAG-систем у реальних ІС провайдерів ІКП. Розпочнемо з описання загального підходу до побудови і дослідження мультимодальної агентної RAG системи.

Мультимодальна агентна RAG-система для інтегрованого оброблення текстових, табличних і графічних даних побудована на архітектурі ReAct-агента (LangChain). Для зберігання ембеддингів використовується векторне сховище ChromaDB [99]. Створення ембеддингів і генерацію відповідей реалізується OpenAI API (text-embedding-ada-002, GPT-4) [100, 101]. Витягування релевантних даних із великих PDF-документів демонструється на прикладі відомих бібліотек PyMuPDF [102] і Camelot-py [103]. Вилучення/виокремлення небажаних і токенізація адекватних блоків тексту здійснюється за допомогою технології spaCy [104]. Особливості оброблення блоків тексту і переведення таблиць у текстовий формат будуть деталізовані при описанні роботи системи.

Для досягнення цілей дисертаційного дослідження, реалізації здатностей інтелектуального асистента важливо забезпечити створюваній системі здатність виконувати багатокрокове логічне виведення. Для цього агент повинен поетапно планувати дії, звертатися до різних компонентів TextSearch, ImageSearch, TableSearch і інтегрувати отримані результати у фінальну відповідь. Отже, необхідно у мультимодальному контексті реалізувати агентну логіку для пошуку та генерації відповідей. У плані реалізації логіки спочатку розробимо алгоритм вибору модальностей, виходячи із змісту запиту. Потім напрацюємо стратегію багатокрокового логічного виведення для уточнення проміжних результатів. Залишається агрегувати результати опрацювання різних модальностей у єдину відповідь користувачеві. Оцінювати ефективність запропонованої RAG-системи будемо у порівнянні з текстовою за загально прийнятими показниками Coverage, Accuracy, Recall, Latency, User Relevance.

Наступним кроком дослідження є вибір і характеристика даних, які використовуються для проведення експериментального дослідження.

Першу перевірку і оцінювання запропонованої мультимодальної агентної RAG-системи виконаємо на основі “Global Tuberculosis Report 2024” Всесвітньої організації охорони здоров’я (повний файл основного звіту доступний на офіційному сайті організації – ISBN 978-92-4-010153-1, дата публікації – 29 жовтня 2024 р.) [97].

Текстові розділи (більше 25 000 слів), статистичні таблиці (18 шт.) та ілюстративні матеріали (карти, гістограми, діаграми – 34 шт., фотоматеріали – 6 шт.) звіту дозволять тестувати всі модальності системи. Представлені дані країн (статистична інформація збирається щорічно з 193 країн), глобальні та регіональні дані, сценарії розвитку ситуації, динаміки за 2000-2023 рр. вимагають від системи здатності до багатокрокового логічного виведення.

В цілях експерименту в документі виділено такі три модальні колекції:

- text_collection (абзаци та підписи до рисунків);
- table_collection (кожен рядок таблиці збережений окремим документом – конкатенація клітинок через символ «|»);
- image_caption_collection (текстові описи графіків).

Отже для тестування можливостей мультимодальної агентної RAG-системи маємо достатню кількість унікальних, структурно різнорідних даних.

Тепер опишемо архітектуру системи та її компоненти.

Розпочнемо з того, що коротко опишемо принцип роботи системи, яка реалізує парадигму мультимодальної RAG-системи. Сутність систем цього класу визначає підсилення генеративних можливостей LLM за рахунок попереднього формування семантичного опису запитання. RAG-система спочатку здійснює семантичний пошук у векторному сховищі (етап retrieval) передає знайдений контекст у запиті до компоненту LLM, який формує відповідь на запит. З врахуванням поділу векторного сховища на колекції векторів трьох модальностей – текст, таблиці, графічні описи – в роботі мультимодальної RAG-системи можна виділити такі ключові етапи:

1) реалізація ітераційного процесу: а) пошук у описаних вище колекціях векторів (query embedding) – б) аналіз запиту і результатів пошуку – в) додаткова спроба уточнення у тій же колекції (за необхідності) – д) перехід до іншої модальності;

2) трансформація контексту (Top-k документів для кожної модальності) у запит (aggregation);

3) формування відповіді компонентом LLM (наприклад, GPT-4).

На першому етапі основними концепціями є агентна взаємодія та багатокрокове логічне виведення. Базовою парадигмою є описаний вище шаблон ReAct як основа агентної взаємодії з огляду на:

- декларативну прозорість процесу міркувань (ReAct вимагає від LLM реалізувати міркування за рекурсивною схемою «думка → дія → спостереження», що зменшує вірогідність галюцинацій завдяки підтримці кожного твердження конкретним спостереженням і можливості експертного аудиту й відтворюваності ланцюжка міркувань);
- підтримку уточнювальних підзапитів (self-ask) та багатокрокового планування без сценаріїв притаманних скінченим автоматам (згадана вище рекурсивна схема надає моделі здатність ініціювати уточнювальні підзапити та будувати ланцюжки з багатьох спроб без необхідності жорсткого скриптового керування притаманного скінченим автоматам, що є важливим для мультимодального середовища, у якому формування відповіді вимагає інтеграції інформації з різних джерел);
- повну телеметрію агентних дій (інтеграція ReAct із механізмом зворотних викликів LangChain забезпечує автоматичний запис кожної реалізації рекурсивної схеми у форматі JSON, що дозволяє виконувати кількісну та якісну аналітику, наприклад, визначати середню кількість hops, оцінки часу на кожну дію, пошук «вузьких місць»);
- пристосованість до великого контекстного вікна GPT-4 (ці моделі здатні утримувати та обробляти підказки обсягом до 128 тисяч токенів, що робить ефективною явну вербалізацію кроків міркувань і мінімізує втрату

інформації між ітераціями).

Важливим для першого етапу також є мультимодальний семантичний пошук. Гетерогенний характер вхідних даних (текст, таблиці, графічні матеріали) обумовив реалізацію пошукового модуля за сукупністю трьох незалежних векторних індексів. Це дозволяє врахувати специфіку розподілу ознак у цих векторних просторах, сформованих моделями Sentence-BERT [105] (для тексту і рядків таблиць) та CLIP (для описів графіків):

1) в text_collection кожний абзац тексту (після токенізації та нормалізації) засобами API text-embedding-ada-002 перетворюється на 1536-вимірний вектор;

2) в table_collection кожний рядок таблиці конкатенується через роздільник «|» та векторизується на основі тієї ж моделі, проте зберігається в окремому HNSW-індексі (експериментально встановлено, що поділ індексів зменшує середній час пошуку найближчого сусіда (nearest-neighbor) на ≈17 % порівняно з монолітною схемою);

3) в image_caption_collection описи графіків і alt-теги проходять попередню лематизацію, усунення stop-слів і векторизацію Sentence-BERT (MiniLM-L12)-моделлю, оптимізованою для коротких рядків (текстова репрезентація капшенів вигідніша, оскільки використання CLIP-ембеддингів без додаткових GPU-ресурсів на локальній машині вимагає ≈220 мс/запит).

Алгоритм пошукової сесії:

Крок 1. Формування запиту до векторної БД (query embedding).

Крок 2. Надсилання запиту паралельно до всіх трьох колекцій з параметром $k = 6$.

Крок 3. Сортування результатів за косинусною відстанню та об’єднання у спільну чергу пріоритетів (для опрацювання вибираються top-k = 10 документів з урахуванням емпірично визначеного відношення текст: таблиця: графіка. Виявилося, що саме вибір елементів цих колекцій у відношенні 6:2:2 забезпечує мінімальну довжину prompt-контексту без втрати повноти (Recall).

Важливе значення для ефективності векторного пошуку має вибір найбільш придатної кількості документів, які повертаються як результат пошуку. У роботі на основі оцінювання балансу між повнотою і ефективністю (Latency/Token Cost) емпірично обґрунтовано, що найкраще рішення досягається при значенні $k = 6$. Обґрунтування наведене у додатку Б.

Розглянемо детальніше компоненти RAG – пошуковий і генеративний модулі та база знань, їх призначення, функції і особливості реалізації:

1) *Пошуковий модуль (Retriever)* – здійснює пошук релевантних документів і даних у базі знань у два кроки: спочатку перетворює запит користувача у векторне представлення; потім на основі семантичної схожості шукає релевантні документи і дані у базі знань. Інструментами реалізації модуля є векторні БД, насамперед Pinecone, Chroma або Milvus і моделі embeddings, насамперед Sentence-BERT або OpenAI embeddings;

2) *Генеративний модуль (Generator)* – створює текстові відповіді на основі знайдених пошуковим модулем даних у два кроки: інтеграція результатів пошуку; генерація відповідей з урахуванням контексту та стилю користувача. Модуль реалізується з використанням великих мовних моделей, насамперед GPT-4, LLaMA 2, Anthropic Claude. Для навчання генеративної моделі використовуються дані, надані пошуковим модулем;

3) *База знань (Knowledge Base)* – містить структуровану інформацію, яка використовується для пошуку. Нефункціональними вимогами до база знань є висока якість даних, релевантність даних до потреб системи, оптимізована під потреби швидкого доступу та індексації структура. Джерела даних – технічна документація, бізнес-звіти, історичні дані, специфікації тощо. Для індексації документів бази знань використовуються векторні представлення, а для зберігання документів – векторні бази, насамперед Pinecone, Milvus.

Кожен компонент RAG-системи виконує чітко визначену функцію. Основні компоненти потоку даних системи: формулювання запитів; інтеграція отриманих результатів пошуку за текстовими запитами користувача; перетворення запиту у векторне представлення; пошук релевантних

документів у базі знань пошуковим модулем; передача пошуковим модулем знайдених документів; генерація текстової відповіді, узгодженої із запитом користувача; надання користувачеві відповіді через інтерфейс системи.

Кроки інтеграції RAG в платформу підтримки ЖЦ сервісів наведені у додатку Б. З інтеграцією RAG в платформу підтримки ЖЦ сервісів пов'язані такі виклики: низька якість даних у базі знань; значна затримка пошуку у великих базах; обмеження генеративної моделі у розумінні контексту. Ефективною реакцією на зазначені виклики є регулярне оновлення та очищення бази знань, використання кешування для повторюваних запитів і адаптація генеративної моделі до вузької доменної області.

Щоб підтвердити правильність замислу, проведемо дослідження.

Для проведення дослідження потрібні надійні дані. Процеси підготовки даних та генерації бази даних, які описані у додатку Б, включають: вибір необхідних даних із PDF документа; генерація векторів; зберігання отриманих векторів; перенесення графічної інформації до векторної БД.

*Організація мультимодального пошуку*

Механізм пошуку реалізовано як трирівневу систему, де кожна модальність – текст, таблиці та лематизовані підписи до зображень – обслуговується введеною вище окремою векторною колекцією ChromaDB, а взаємодія з ними здійснюється через обгортки LangChain – TextSearch, TableSearch і ImageCaptionSearch. Модель FastText швидко перетворює кожен вхідний запит у числовий вектор навіть для незнайомих слів.

Вектор порівнюється з трьома наперед підготовленими «якірними» (англ. anchor) векторами, які відповідають поняттям text, table та figure. По суті це еталонні вектори, що «уособлюють» кожну модальність. Ми вимірюємо косинусну відстань між запитом і кожним якорем; чим менша відстань (тобто чим більший показник схожості), тим ближчий запит до цієї модальності.

Якщо найбільший показник схожості опускається нижче 0,35 (це поріг, підібраний експериментально), система робить висновок, що запит змішаний і

опрацьовує і текстові, і табличні дані. Такий пошук забезпечує вищу повноту, особливо у запиті містяться текстова частина і числові дані таблиць [106].

Якірні вектори (“text”, “table”, “figure”) створюються під час старту системи: для кожного слова-якоря обчислюється 300-вимірний вектор. Оскільки їх лише три, тримати їх у векторній БД недоцільно; вони зберігаються як звичайний словник Python у модулі, що відповідає за класифікацію запиту. Під час старту системи цей словник завантажується в оперативну пам’ять, і далі при кожному запиті обчислюється косинусна відстань між вектором запиту та цими трьома еталонними векторами.

Запити до кожної модальності виконуються паралельно з фіксованою глибиною $k = 6$. Експерименти показали, що це забезпечує оптимальний баланс між повнотою та обсягом контексту. Після отримання кандидатів результати сортуються за косинусною відстанню та збираються у спільний пул, з якого формується підсумковий контекст top-10 документів у пропорції 6 : 2 : 2 (текст : таблиці : графіка). Отже, система поєднує спеціалізовані пошуки в однорідних просторах із єдиним механізмом ранжування, що мінімізує затримку та підтримує високий рівень повноти відповіді.

*Агентна логіка та багатокрокове логічне виведення*

Модуль прийняття рішень реалізовано у форматі ReAct-агента (AgentType.CHAT_CONVERSATIONAL_REACT_DESCRIPTION) на основі GPT-4 із нульовою температурою та лімітом 700 вихідних токенів, що гарантує детермінованість та достатнє вікно контексту для мультимодальної відповіді. Після кожного спостереження виконується внутрішня перевірка: якщо релевантність отриманих фрагментів до запиту, виміряна як середня косинусна подібність ключових токенів, нижча порогу 0,75, автоматично виконується додатковий крок пошуку шляхом синонімічного перефразування з використанням словника WordNet. Тобто реалізується механізм self-ask та багатокрокового планування без жорсткої скриптової логіки.

Число ітерацій логічного виведення обмежено трьома: якщо після першого hop-у система не досягає повної покритості фактів (Recall < 1.0), агент

повертає часткову відповідь, чітко позначаючи пропущені елементи маркером <missing>. Під «пропущеними елементами» маються на увазі факти або фрагменти контексту, які необхідні для повної й коректної відповіді, але не були знайдені навіть після трьох ітерацій пошуку. Це можуть бути, наприклад, числові показники з таблиці чи ключові формулювання з тексту, без яких висновок буде неповним. Якщо після третьої спроби система все ще не знаходить усіх потрібних фрагментів (що фіксується як Recall < 1.0), агент повертає «часткову» відповідь і вставляє у відповідному місці маркер <missing>. Маркер сигналізує рецензенту або кінцевому користувачеві, що в базі знань не вдалося відшукати конкретну інформацію, тож наведена відповідь може вимагати додаткової верифікації чи розширеного джерела даних. Усі етапи циклу – Thought, Action, Observation – автоматично журналюються засобами callback-менеджера LangChain у форматі JSONL (каталог logs/<run_id>.jsonl), що забезпечує детальний аудит та можливість подальшого кількісного аналізу ефективності процесу логічного виведення.

Наведемо отримані в процесі проведення дослідження результати.

Порівняння отриманих результатів з найбільш популярними спробами реалізації можливостей RAG-систем показало переваги запропонованого підходу. Так, у системі Active RAG [86] за рахунок навчання пошуковий вдало вибирає мінімальний підкорпус документів, забезпечуючи заданий рівень точності відповіді і високу ефективність. Однак, при цьому система є моно-модальною і не підтримує багатокрокове планування дій. Система REPLUG [87] за рахунок інтерактивного повторного використання (plug-in) уже знайдених фрагментів знань надає LLM здатність формувати уточнювальний запит, отримувати нові документи і ефективно використовувати у наступній ітерації генерації для підвищення точності відповідей. Але REPLUG має той же недолік, обмежуючись текстом. Відомі системи Retro [90], Toolformer [91] також є мономодальними, базуються на статичному вбудовуванні знань.

## 2.5 Побудова архітектури сервісів на основі нейромережі з глибоким навчанням

У ЖЦ сервісів провайдерів ІКП важливе місце посідає етап проєктування сервісу. Тут здійснюється побудова архітектури сервісу. Архітектори сервісів визначають компоненти сервісу і їх зв'язки з врахуванням визначених функціональних і нефункціональних вимог до сервісу, наявних обмежень і ризиків. Навіть архітектори, які мають високу кваліфікацію і накопичений досвід, витрачають багато часу і зусиль на побудову архітектури сервісів. Помилки, які трапляються, та зміни умов надання сервісів вимагають перепроєктування сервісів. Тому важливо автоматизувати процес побудови архітектури сервісів. Існуючі підходи не враховують усього комплексу чинників впливу на побудову архітектури. Створення сучасної точної і ефективної ІТ побудови архітектури сервісів постає як нагальна потреба. Видається, що таку технологію можна створити на основі нейронної мережі. Впровадження такої технології дозволить будувати архітектури з врахуванням різних критеріїв, насамперед ефективності, масштабованості, розвитку. Це закладе основу для ефективних рішень на етапах реалізації, надання сервісів та ін. У підрозділі розглядаються основи побудови архітектури сервісів з використанням нейромереж.

### 2.5.1 Потреби, передумови і результати побудови архітектури сервісів

Оскільки для галузі інфокомунікацій сервісний підхід є традиційним і постійно вдосконалюється, вибір інструментів його впровадження постійно розширюється і оновлюється [107]. Не є винятком інструменти підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП [108, 109]. Це обумовлено переходом до створення ІС провайдерів ІКП за моделлю E2E. Постала проблема побудови архітектури сервісів в ІС провайдерів ІКП, інспірована впровадженням концепції архітектури в сучасних методологіях створення ІС [110–112]. Проєктування сервісу полягає у побудові його архітектури. З

врахуванням сучасних концепцій ІТ-галузі і галузі інфокомунікацій побудова архітектури сервісів перетворилася в складну проблему, набула нового змісту. Технології її побудови базуються на використанні графічних мов, специфічних для кожної ІТ-компанії і провайдера ІКП контексту і шаблонів.

Складна задача побудови архітектури сервісу очікує вирішення на основі формальної моделі і відповідного методу. Автоматизація побудови архітектури сервісу базується на методах та технологіях ШІ [113]. Сьогодні інструменти ШІ, зокрема технології машинного навчання та нейронні мережі, широко застосовуються у задачах з комп'ютерного зору [114], розуміння і генерування природномовних текстів [115, 116], трансформації звуку в текст і навпаки [117], генерування зображень і відео [118], автоматичного керування і прийняття рішень [119, 120] та ін. Аналіз джерел показав, що універсальні апроксимаційні властивості, здатність до навчання та інші базові властивості нейронних мереж дозволяють вирішувати широкий клас задач з високою точністю. Якщо формалізувати вхідні та вихідні дані задачі, нейронну мережу можна навчити відображати зв'язок між ними на основі зразків вчителя. Ще кращі результати сьогодні забезпечують нейромережі з глибоким навчанням. Отже виправданим буде підхід до побудови архітектури сервісів, які надаються провайдерам ІКП в ІС ІТ-компаній-розробників за моделлю E2E чи провайдерами ІКП своїм клієнтам, за вимогами до даних сервісів.

Розглянемо проблему глибше. Надаючи клієнтам сервіси, провайдеру ІКП необхідно мінімізувати витрати, щоб пропонувати вигідні умови на ринку [63, 64]. Провайдер надає клієнтам комплекси сервісів, як і ІТ-компанія надає комплекси сервісів провайдерам. Враховуючи потребу надання сервісу в рамках існуючої інфраструктури провайдера, загальне рішення зі структури інтегрованого сервісу не видається простим. Беручи до уваги комплекс чинників впливу на це рішення, побудову архітектури сервісу провайдера як структури інтегрованого сервісу не можна назвати простою проблемою.

Проблему побудови архітектури сервісу ускладнюють додаткові чинники, зокрема надання провайдеру сервісу за моделлю E2E. Як проєкти ІС

провайдерів ІКП, розробка, проєктування, реалізація, впровадження і підтримка яких здійснюється на основі моделі E2E, доцільно застосовувати загальну схему архітектури сервісу, яка визначає множину усіх можливих архітектур сервісів. Тоді компонентами архітектури для конкретного провайдера будуть елементи таких класів: доменні високорівневі схеми; доменні схеми компонентів; доменні компоненти (успадковані компоненти провайдера, компоненти ІТ-компанії, компоненти третіх сторін); компоненти аналітики, ШІ і підтримки прийняття рішень; засоби взаємодії і комунікації; технічні засоби; засоби роботи з даними. При цьому компоненти ІТ-компанії та третіх сторін мають відповідати стандартам TM Forum, що не стосується успадкованих компонентів провайдера.

Тоді проблема полягає у тому, щоб на основі інформації про провайдера згідно зазначеної схеми побудувати архітектуру ІС провайдера (архітектуру сервісу), яка оптимальним чином відповідає його вимогам і очікуванням. Побудовану для провайдера архітектури сервісу назвемо персоналізованою.

Розглянута проблема будемо вирішується у два етапи. На першому етапі будується попередня архітектури сервісу, яка передається провайдеру для аналізу, можливих змін і узгодження. На другому етапі будується остаточна архітектура сервісу, яка буде затверджена і покладена в основу реалізації.

Описані етапи реалізують різні нейромережі. Їх викликає через REST API структурована LLM, яка знає особливості ув’язки і підтримки процесів ЖЦ сервісів в ІС провайдерів, створених за моделлю E2E. Наведемо загальний опис пропонованого рішення, яке полягає у створенні інтелектуальної ІТ побудови архітектур сервісів на основі нейромереж. Створення інтелектуальної зазначеної ІТ передбачає виконання таких кроків:

1) аналіз вхідних та вихідних даних, збір даних для навчання мережі;
2) формалізація вхідних і вихідних даних для нейронної мережі;
3) розроблення моделі нейромережі для генерації архітектур сервісів;
4) розроблення методу навчання моделі генерації архітектур сервісів;

5) розроблення технології автоматизованого проєктування архітектур сервісів на базі запропонованої нейромережевої моделі.

Отже, вибір типу нейронної мережі, її структурування, підбір алгоритмів функціонування і взаємодії структурних елементів і навчання нейромережі складуть основу побудови архітектури сервісів. Отримуючи на вході характеристики провайдера ІКП і його вимоги до ІС, нейромережа генерує архітектуру сервісу. Інспірований концепцією ЖЦ сервісів цей підхід передбачає тісний зв'язок і взаємне використання результатів етапів бізнес-аналізу, інженерії вимог і проєктування. При навчанні нейромережі на вхід подаватимемо підготовлені і закодовані дані етапів бізнес-аналізу, інженерії вимог разом з закодованими представленнями відповідних архітектур сервісів. В основу вибору типу, структурування і методу навчання нейронної мережі покладемо класифікацію вхідних даних і компонентів архітектури навчальних зразків і пошук спільних елементів відображення перших у другі.

Перейдемо до аналізу вхідних, вихідних даних та їх формалізації. Спочатку визначимо дані, що подаються на вхід і отримуються на виході мережі. Потім сформуємо повний набір даних. Для сприйняття даних мережею розробимо систему їх формального представлення і кодування.

Вище було висловлене припущення, що на побудову архітектури сервісу будуть впливати дані провайдерів і клієнтів, оцінки їх діяльності, вимоги до сервісів. Експериментально встановлено, що з точки зору точності найбільш придатним є подання на вхід нейронної мережі значень шести груп параметрів, описаних у додатку Д. Опишемо більш детально специфіку їх впливу.

Загальні параметри бізнес-діяльності провайдерів ІКП (група 1) визначаються на етапі бізнес-аналізу і характеризують цілі, організацію, послуги, методи праці, стейкхолдерів. Експериментально підтверджено їх вплив на вибір елементів архітектури, які забезпечують виконавців ІТ-компанії і провайдера інструментами аналізу, прогнозування, оцінювання, а також визначають їх зв'язок з функціональними елементами. Кількісні характеристики сервісів, що надаються провайдером ІКП за підтримки ІТ-

компанії, та клієнтів (група 2) впливають на вибір елементів архітектури, які забезпечують умови, за яких функціональні елементи архітектури через їх взаємодію, управління доступом до ресурсів забезпечать бажану поведінку ІС провайдера ІКП.

Функціональні вимоги (група 3), серед яких виділяються головні функції і функції в рамках головних, визначають вибір компонентів архітектури (перші), їх функціональність і вибір додаткових функціональних компонентів (другі). Нефункціональні вимоги (група 4), типовим прикладом є параметри SLA, впливають, по-перше, на вибір функціональних компонентів архітектури, по-друге – на вибір нефункціональних компнентів архітектури, які забезпечують умови, за яких функціональні елементи архітектури через їх взаємодію, управління доступом до ресурсів забезпечать ІС провайдера ІКП бажану поведінку.

Специфічні вимоги до сервісів, які надаються провайдером ІКП (група 5) впливають, з одного боку, на вибір функціональних елементів, які забезпечать ІС провайдера бажану поведінку, з іншого боку, вони впливають на додавання компонентів третіх сторін. Додаткові вимоги провайдера ІКП до сервісів (група 6) впливають на додавання до архітектури додаткових функціональних і нефункціональних компонентів ІТ-компанії та компонентів третіх сторін.

Формально представимо вимоги до проєктування сервісів як кортеж:

$$SARP = (N, G, Q, F, nF, S, A, M). \quad (2.1)$$

Тут використовуються такі позначення: *SARP* – ідентифікатор проєкту; *N* – ідентифікатор вимог; *G, Q, F, nF, S, A* – відповідно значення шести груп вимог, *M* – матриця архітектури сервісу.

В залежності від типу параметра змінні можуть приймати як бінарні, так і кількісні значення. Якщо вимога до архітектури сервісу по характеристиці має бінарний характер, то змінна приймає значення 0, якщо вимога відсутня, або 1, якщо вимога присутня. Якщо вимога має кількісних характер, вона приймає значення від 0 до 1, 0 – вимога відсутня та діапазон від 1 до 100%, що

описує кількісні характеристики вимоги в відсотковому відношенні від мінімальних вимог до максимальних. У формальному представленні вимоги до архітектури сервісу перетворюються на одномірний масив. Кожний елемент масиву відображає блок та змінну даного блоку. Схематично подання вимог до архітектури сервісу представлено формулою (2.2):

$$N1 = \left( \underbrace{\underbrace{0}_{G_{1,1}}\ \underbrace{1}_{G_{1,j}}\ \underbrace{0}_{G_{2,1}}\ \underbrace{0.51}_{G_{1,j}}\ \underbrace{0.72\ 0.42\ 0.55\ 0.23}_{G_{3,j}}}_{G_{ij}}\ \underbrace{\ldots,\ldots,\ldots,\ldots,\ldots,\ldots}_{Q_{ij}\ F_{ij}\ nF_{ij}\ S_{ij}}\ \underbrace{0\ 1\ \ 0}_{A_{11,1}\ \ A_{ij}} \right) \quad (2.2)$$

У формулі (2.2) використовуються такі позначення:

$N_1$ – ідентифікатор вимог (проєкту);

$G_{ij}$ – вимоги до загальних характеристик (містять 12 блоків змінних $G_i$ по декілька змінних в кожному блоці $G_{ij}$, відповідно $G_1$, 1 = 0 – перша змінна з бінарним значенням, першого блоку загальних характеристик – клас провайдера за системою класифікації, $G_2$, 1 = 0.51 – перша змінна з кількісним значенням, другого блоку загальних характеристик – кількість користувальницьких сервісів провайдера і т.д.);

$Q_{ij}$ – вимоги до кількісних характеристик сервісу, архітектура якого будується (містять 3 блоки змінних $Q_i$, кожен від однієї до 15 змінних $Q_{ij}$);

$F_{ij}$ – функціональні вимоги до сервісу, архітектура якого будується (містять 19 блоків змінних $F_i$, що відповідають вимогам до наявності функціонального блоку в архітектурі сервісу, по декілька змінних в кожному блоці $F_{ij}$, кожна змінна відповідає функціям даного блоку);

$nF_{ij}$ – нефункціональні вимоги до сервісу, архітектура якого будується (містять 10 блоків змінних $nF_i$, кожен блок містить декілька змінних $nF_{ij}$);

$S_{ij}$ – специфічні вимоги до сервісів (містять 5 блоків змінних $S_i$, кожен блок має по декілька змінних $S_{ij}$);

$A_{ij}$ – додаткові вимоги (містять 12 блоків змінних $A_i$, кожен блок має від однієї до декількох змінних $A_{ij}$).

Для визначення усіх можливих архітектур сервісу необхідно описати відповідний клас схем, який назвемо загальною схемою архітектури сервісу – *M*. Традиційне представлення архітектури систем, їх програмного забезпечення, а тепер і сервісів, базується на їх поділі на компоненти і визначенні взаємодії компонентів, яку подаємо їх взаємозв'язками. Зрозуміло, що компоненти сервісів, як і їх взаємозв'язки, в галузі інфокомунікацій мають свою класифікацію і специфічні особливості.

Архітектура сервісів звичайно представляється графічними мовами і подається у вигляді графа, вершини якого представляють компоненти, а ребра – їх взаємозв'зки. Для подання графів у зручному для аналізу вигляді використовуються матриці. Надалі будемо використовувати квадратну матрицю $M_{nxn}$, загальний вигляд якої наведений у таблиці 2.1.

Таблиця 2.1 – загальний вигляд матриці подання архітектури сервісів

| | $\boldsymbol{K_1}$ | … | $\boldsymbol{K_j}$ | … | $\boldsymbol{K_n}$ |
|---|---|---|---|---|---|
| $\boldsymbol{K_1}$ | $m_{11}$ | … | $m_{1j}$ | … | $m_{1n}$ |
| … | … | … | … | … | … |
| $\boldsymbol{K_i}$ | $m_{i1}$ | … | $m_{ij}$ | … | $m_{in}$ |
| … | … | … | … | … | … |
| $\boldsymbol{K_n}$ | $m_{n1}$ | … | $m_{nj}$ | … | $m_{nn}$ |

Тут $K_i$, $i$=1,…,$n$, позначають компоненти, які використовуються для побудови архітектури сервісів. Інакше кажучи, $K_i$, $i$=1,…,$n$, можна розглядати як вершини графа, який репрезентує архітектуру сервісу. Для матриці, яка представляє ту ж архітектуру сервісу, кожному $K_i$, $i$=1,…,$n$, відповідає рядок і стовпець. Тоді зв'язкам компонентів $K_i$ та $K_j$ в архітектурі сервісу або ребру, яке поєднує вершину $K_i$ з вершиною $K_j$ графу, який репрезентує архітектуру, в матриці відповідає елемент, який позначимо $m_{ij}$. Цей елемент на перетині рядка $i$ та стовпця $j$ визначає використання компонентом $K_i$ результатів роботи компонента $K_j$. Елемент $m_{ij}$ матриці $M_{nxn}$ може набувати таких значень:

1) опис параметрів запиту, який компонент $K_i$ надсилає компоненту $K_j$, і опис даних, які компонент $K_i$ отримує у відповідь на запит від компонента $K_j$;

2) 0, якщо компонент $K_i$ не використовує компонент $K_j$.

Отже, квадратна матриця $M_{nxn} = \| m_{ij} \|$ рядки задає архітектуру сервісу і, за потреби, може бути представлена схематично за допомогою графічних мов.

Специфічними для архітектури сервісів є компоненти, які поділяються на: успадковані компоненти провайдера ІКП; компоненти ІТ-компанії; компоненти третіх сторін; компоненти аналітики; компоненти ШІ і підтримки прийняття рішень; засоби взаємодії і комунікації; технічні засоби; засоби роботи з даними та ін. Кожному компоненту з множини $K$ приписаний порядковий номер, від 1 до 21. Є відповідність між номерами компонентів і номерам рядків та стовпців матриці $M_{nxn}$. Конкретизуємо формат ненульових елементів матриці для відображення характеру взаємозв'язків відповідних компонентів архітектури сервісу: 1 – зв’язок присутній і обов’язковий; 2 – зв’язок присутній, але опційний. Приклад закодованої архітектури сервісу в ІС провайдерів ІКП надає формула (2.3):

$$M_{nxn} = \begin{pmatrix} 0 & 1 & \dots & 2 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \tag{2.3}$$

За наявності реально вживаної при проектуванні кількості типів компонентів розміри матриці $M_{nxn}$ обмежимо 21 стовпцем і 21 рядком. Нульова головна діагональ представленої формулою (2.3) матриці свідчить про відсутність зв’язків компонентів самих з собою. Прикладами обов’язкового і опційного зв’язків є зв’язки між компонентами $K_1$ та $K_2$ ($m_{1,2} = 1$) і між компонентами $K_1$ та $K_{21}$ ($m_{1,21} = 2$) архітектури відповідно.

Наведені вище формальні конструкції для представлення вимог до сервісів, необхідні для побудови їх архітектури (див. формулу 2.1) та самої архітектури сервісу (див. формулу 2.3) використаємо як формальну основу для трансформації, яку здійснює, відповідно кодувальник вхідних даних для нейронної мережі і декодувальник її вихідних даних в процесі побудови архітектури сервісів в ІС провайдерів ІКП. Доцільно такий підхід

використовувати в процесі навчання нейронної мережі здатності будувати архітектуру сервісів на основі формалізованих представлень вимог до них. Перейдемо до вибору типу мережі для побудови архітектури сервісів.

### 2.5.2 Обгрунтування вибору нейромережі для реалізації підходу до побудови архітектури сервісів

Обгрунтований вибір типу нейронної мережі, призначеної для побудови архітектури сервісів, дозволяють зробити результати аналізу використання нейронних мереж для розв’язання різноманітних прикладних задач, наведений у додатку Б. Встановлено, що на вибір нейронної мережі для побудови архітектури сервісів впливають такі чинники: цільове призначення (розпізнавання зображень, генерування зображень,…); способ відображення вхідних даних у вихідні або модель (дифузійна, трансформер,…); наявність опрацьованих технологій навчання; характер і обсяг вхідних і вихідних даних; додаткові вимоги (нефункціональні та інші).

Перший крок, визначений цільовим призначенням нейронної мережі, спрямованої на побудову архітектур сервісів, вказав на доцільність вибору генеративних нейромереж. Дійсно, архітектура сервісу формалізовано подається графічними мовами, які мають візуальну інтерпретацію. Це можна інтерпретувати генеруванням зображень, де високу ефективність демонструють генеративні нейромережі [117, 118]. Тому для нейромережі побудови архітектури сервісів доцільним є вибір генеративних нейромереж.

Вибір моделі нейронної мережі визначається характером і обсягом вхідних і вихідних даних, способом відображення перших у другі. На вхід нейронної мережі вхідні дані, визначені формулою (2.1) і представлені у закодованому вигляді, описаному формулою (2.2). На виході нейронної мережі отримуємо матрицю архітектури сервісу, представлену формулою (2.3). Тобто йдеться про відображення спільних фрагментів вхідних даних у структурні компоненти і параметри їх обміну інформацією. Тут придатними з точки зору характеру відображення і точності виявилися дифузійні моделі

[121] та трансформери з мультиувагою [122–124]. Врахування інших чинників впливу схилило вибір на користь моделі трансформер.

Структура і взаємозв'язки компонентів нейронної мережі побудови архітектури сервісу представлені на рис. 2.2. Вона легко інтегрується з іншими компонентами платформи підтримки ЖЦ сервісів, покладеної в основу ІС провайдерів ІКП. Крім того, вона підтримує усі кроки побудови архітектури сервісів: 1) збір і введення вимог до сервісу; 2) кодування вимог до сервісу токенізатором вимог з трансформацією вхідних даних в тензори, які подаються на вхід нейронної мережі; 3) генерування нейронною мережею тензору – виходу трансформера; 4) декодування архітектури сервісу і її візуалізацію; 5) надання архітектору можливості для аналізу архітектури і її опрацювання з візуалізацією впливу описаних вище чинників.

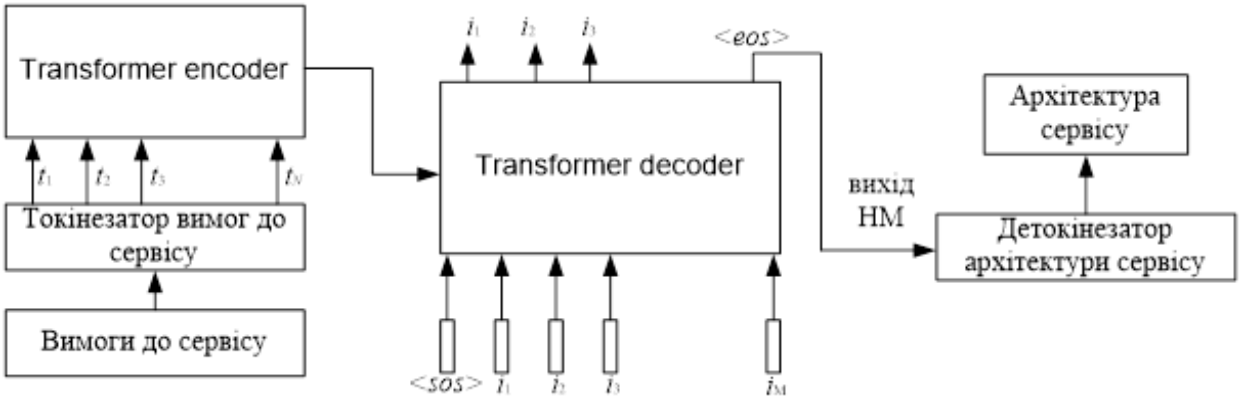


Рисунок 2.2 – Структура і взаємозв'язки компонентів нейронної мережі побудови архітектури сервісу

Обґрунтувавши рішення щодо структури і взаємозв'язків компонентів нейронної мережі побудови архітектури сервісу перейдемо до її навчання.

### 2.5.3 Навчання нейронної мережі

Для реалізації усіх описаних вище кроків побудови архітектури сервісів нейронна мережа має бути попередньо навчена з урахуванням специфіки провайдера ІКП. Будемо виконувати навчання нейромережевої моделі на основі відомого підходу навчання з учителем. Реалізація окресленого підходу до навчання побудована на тому, що визначаються спільні фрагменти

відображення параметрів вимог у компоненти і зв’язки нейронної мережі у парах вхідних (вимоги до сервісу) і вихідних (відповідна архітектура сервісу) даних, які подаються на вхід нейронної мережі у процесі навчання. На основі сформованих відповідностей навчальні запити трансформуються у прогнозовані відповіді, які використовуються для оцінювання точності мережі. Якщо отримана оцінка нижче встановленого порогу, навчання продовжується. Тут важливо підготувати повний набір розмічених даних, який враховує вплив усіх чинників. Навчання моделі полягає у циклічному виконанні до заданого рівня точності такої послідовності кроків:

Крок 1. Подання на вхід мережі навчального набору даних, який складається з пар «закодовані вимоги до сервісу, які подаються на вхід, закодована архітектура, яку нейронна мережа повинна згенерувати на виході».

Крок 2. Реалізація алгоритму навчання. Призначення алгоритму навчання полягає у підборі таких параметрів нейромережевої моделі, при яких буде досягатися визначений рівень точності;

Крок 3. Оцінювання точності навчання: обчислюємо мінімальну сумарну середньоквадратичну похибку між поточними виходами моделі та виходами, заданими вчителем при тих же заданих вхідних даних, для визначеної кількості запитів;

Крок 4. За необхідності повернення на перший крок;

Крок 5. Візуалізація результатів навчання.

Взаємозв’язки компонентів нейронної мережі побудови архітектури сервісу в процесі навчання представлені на рис. 2.3. Запропонований підхід до навчання ефективно поєднує переваги нейронних мереж типу трансформер для побудови архітектури сервісу, накопичений досвід архітекторів сервісів у галузі інфокомунікацій і сучасні концепції навчання.

Деталізуємо деякі взаємозв’язки компонентів. Вимоги до сервісу кодуються за допомогою формули (2.2). Отримавши на вході матрицю вимог, нейронна мережа на виході формує матрицю архітектури сервісу. Детокенізатор візуалізує архітектури сервісу, реалізуючи процес обернений до

токенізатора. Токенізатор архітектури трансформує архітектури сервісів навчального набору даних в матрицю. Матриця використовується алгоритмом для навчання мережі за стратегією навчання з вчителем. Перетворення архітектури сервісу в матрицю *M* описане таблицею 2.1 та формулою (2.3).

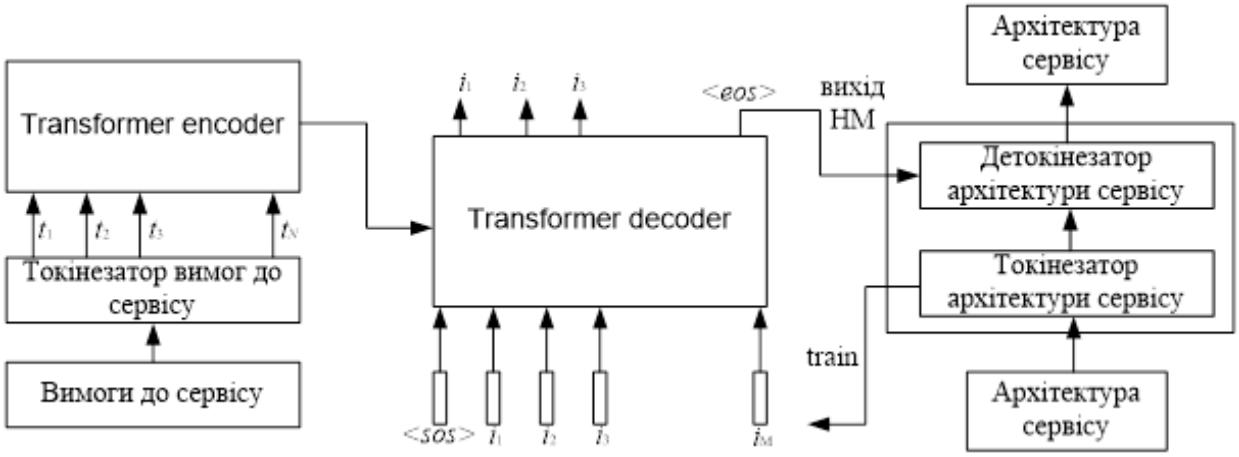


Рисунок 2.3 – Взаємозв’язки компонентів нейронної мережі побудови архітектури сервісу в процесі навчання

У процесі навчання нейронної мережі за поданими на вхід парами «вимоги – архітектура» визначаються спільні характеристики провайдерів ІКП, сервісів, інші спільні вхідні дані, на основі яких підбираються архітектури з відповідними спільними високорівневими схемами, доменними схемами компонентів або однаковими компонентами і взаємозв’язками.

Після навчання нейронна мережа може використовуватися для побудови архітектури сервісів. При цьому доцільно, щоб побудовані архітектури сервісів після уточнення і прийняття архітектором сервісів використовувалися для донавчання нейронної мережі. Це дозволить розширити її можливості врахування нових спільних конфігурацій і, відповідно, забезпечить підвищення відсотку побудованих правильних архітектур сервісів. При цьому доцільно визначити порогове значення кількості побудованих нових архітектур сервісів, при досягненні якого нейронна мережа перенавчається. Іншими словами уточнені і прийняті архітектором сервісів архітектури разом з відповідними вхідними даними додаються до навчального набору даних і повторюється описаний вище циклічний процес навчання моделі.

**Висновки до розділу 2**

Обгрунтовано реалізацію платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП на основі як перевірених концепцій (інтегрованості платформи, трирівневого описання сервісу, контексту і шаблонів описання сервісів, сучасних архітектур, залучення ресурсів третіх сторін, керованих ІТ-послуг, використання повнофункціональної і спрощеної моделей), так і додаткових концепцій для врахування сучасних умов використання сервісного підходу (розширення, класифікація і групування процесів, зворотній зв'язок з бізнесом і використання даних social media, PaaS, використання моделей і практик CMMI, карти процесів, розроблення моделей сервісів і моделей і методів технологій підтримки процесів усіх етапів ЖЦ сервісів, визначення системи сервісів з врахуванням особливостей бізнесу, змін бізнес-середовища, використання гнучких механізмів автоматизації потреб користувачів у явно описані структури сервісів) з орієнтацією на наскрізні процеси, соціальні аспекти ІТ-середовища, підхід Evergreen.

З дотриманням зазначених перевірених і додаткових концепцій в умовах орієнтації на наскрізні процеси розроблена архітектура платформи підтримки процесів етапів ЖЦ сервісів в ІС провайдерів ІКП.

Запропоновано комплекс моделей і методів, призначений забезпечити комунікацію користувачів з ІС провайдера ІКП в термінах розуміння їх проблем, виділення підпроблем, використання відповідних інструментів їх вирішення, інтеграції рішень і їх застосування, який включає LLM і RAG-системи, інтелектуальні агенти та розроблені механізми їх тісної взаємодії.

Запропоновані концепція і підхід до реалізації інтелектуального асистента для служб ІТ-компанії і провайдерів ІКП, здатного підтримувати кожен аспект діяльності, оцінити його вплив на загальні показники діяльності, підказати рішення, контролювати ситуацію, пропонувати нові ідеї, спрямовані на поліпшення бізнесової діяльності.

Обґрунтовано вибір LLM як основи реалізації інтелектуального асистента переважно як інструмента природномовної комунікації користувачів з платформою підтримки ЖЦ сервісів в ІС провайдерів ІКП. Встановлено, що як компонент інтелектуального асистента для служб ІТ-компанії і провайдерів ІКП LLM має опрацьовувати текстові, табличні і графічні дані. Розроблено RAG-систему для розуміння і підтримки вирішення проблем користувачів шляхом впровадження агентного підходу, що дозволило за рахунок попереднього формування контексту підвищити частку правильних відповідей LLM і заклало основу ефективної взаємодії компонентів платформи в процесі вирішення проблем виконавців.

Розроблено метод формування архітектури сервісів, орієнтований на ефективне використання ресурсів із використанням нейромережевих моделей. Розроблено структуру і технологію навчання нейронної мережі, яка дозволяє будувати архітектури сервісів в ІС провайдерів ІКП на основі параметрів діяльності провайдерів і їх клієнтів, функціональних і нефункціональних вимог, інших чинників впливу. Навчена у такий спосіб нейромережа здатна використовувати досвід побудови ефективної архітектури сервісів.

Експериментальне дослідження підтвердило придатність побудованих за допомогою навченої нейронної мережі архітектур сервісів в ІС провайдерів ІКП, їх здатність враховувати накопичений досвід, продемонструвало досить високу точність побудованих архітектур сервісів.

# 3 МОДЕЛІ І МЕТОДИ ЕТАПІВ ІНІЦІАЦІЇ, АНАЛІЗУ ТА ПІДТРИМКИ СЕРВІСІВ

Розглядаються методи, які складуть основу реалізації інструментів платформи підтримки ЖЦ сервісів для процесів етапів ініціації, аналізу і підтримки сервісів. Для групування сервісів у пакети за формальними постановками 1 і 2 запропоноване пристосування відомих статистичного методу і алгоритму кластеризації, широко вживаних у різних галузях. Для розв'язання цієї задачі у постановці 3 удосконалено керований генетичний алгоритм і розроблено ефективний метод розв'язання змішаних задач математичного програмування на основі поєднання сучасних підходів. Перші два побудовані на концепції використання накопиченого досвіду надання сервісів. Інші два метода уможливлюють реалізацію сучасної концепції ведення бізнесу, що полягає у досягненні взаємної вигоди для обох сторін – ІТ-компанії і провайдерів (провайдера і клієнтів). Забезпечується ефективний пошук близьких до оптимальних рішень за критеріями вигоди, яку отримують в ІТ-компанія і провайдери за умови виконання обмежень на ресурси, цінність сервісів, показники SLA та міжсервісні залежності, враховуючи систему знижок до преференційної ціни. Для отримання бізнес-орієнтованих пакетів необхідно враховувати бізнес-цілі і стратегії провайдера сервісів і клієнтів.

Для вибору найбільш вдалого методу групування сервісів у пакети використаємо метод урахування опіній бізнес- і ІТ-менеджерів – важливої складової компонента групування сервісів у пакети розроблюваної платформи.

## 3.1 Статистичний метод групування сервісів у пакети

Накопичені провайдером дані надання сервісів клієнтам уможливлюють використання статистичного підходу. Стандартні статистичні показники – середні значення і відхилення – традиційно використовуються при групуванні сервісів у пакети у різних галузях [125–127]. Підхід виявився придатним для групування сервісів у пакети, які ІТ-компанія надає провайдерам ІПК.

Розпочнемо з пристосування відомого статистичного методу до групування сервісів у пакети для надання і провайдерам, і їх клієнтам. Для цього будемо використовувати наведену у підрозділі 1.4 формальну постановку 1 задача групування сервісів у пакети. Особливості надання сервісів ІТ-компанією для провайдерів ІКП і останніми для своїх клієнтів вимагають спочатку ретельно визначити показники діяльності, які сформують вихідні дані. Для відповідності формальній постановці задачі дані повинні бути подані у вигляді прийнятих у статистиці понять, насамперед вибірок показників прибутку провайдера для кожного сервісу, передбачуваного у пакеті для провайдера чи клієнта. Алгоритм методу має містити відповідні кроки. На основі сформованих вибірок на нступних кроках алгоритму методу мають бути обчислені значення для кожного сервісу необхідних для прийняття рішень показників – звичайно передбачуваного прибутку провайдера за визначений період, ризику, верхньої і нижньої межі. Тут використовуються традиційні статистичні індикатори. Це необхідно виконати для кожного провайдера. Тепер можна описати алгоритм застосування зазначеного методу для групування сервісів у пакети з використанням формальної постановки 1:

Крок 1. Визначення провайдера і періоду, на який планується пакет.

Крок 2. Вибір даних про сервіси, які надавалися вибраному на кроці 1 провайдеру у визначений на кроці 1 період.

Крок 3. Формування на основі результатів другого кроку вибірок показників прибутку провайдера у визначений період для чергового сервісу.

Крок 4. Розрахунок для кожного сервісу описаних вище статистичних індикаторів – передбачуваного прибутку провайдера за визначений період, ризику, верхньої і нижньої межі.

Крок 5. Якщо не всі сервіси розглянуто, повторюємо крок 4.

Крок 6. Розрахунок описаних вище індикаторів для пакету сервісів в цілому з використанням широко вживаних у статистиці формул.

Крок 7. Остаточний висновок ОПР щодо пакету сервісів для провайдера.

Крок 8. Якщо не всі пакети сервісів розглянуто, переходимо на крок 1.

Тепер покажемо як можна врахувати особливостей виконання ключових кроків описаного алгоритму для провайдерів інфокомунікацій.

Успішному виконанню перших двох кроків сприяє можливість зручного використання накопичених у ІС провайдерів даних про надання сервісів у попередні періоди. У ІС провайдерів накопичуються дані про надавані клієнтам сервіси, в тому числі період часу і ціна $y_i$ надання сервісу $S_i, i = 1, \dots, k$. У процесі аналізу діяльності провайдера визначаються їх витрати $z_i$, пов'язані з наданням сервісу $S_i$. Отже, третій крок легко реалізувати в рамках відповідного компоненту групування сервісів у пакети. Подальші обчислення виконуються на основі сформованих на кроці 3 вибірок $X_i$ із прибутків за період $j$ $x_{ij}$, $i \in [1, k]$, $j \in [1, n]$.

Обчислення четвертого кроку для сервісу розпочинаємо за допомогою наведених у додатку В відомих формул (В.1) для визначення середньої кількості клієнтів сервісу за період і (В.2) для визначення прибутку, отриманого від надання сервісу $S_i$ за період $j$. На цій підставі визначаємо передбачуваний прибуток провайдера, ризик, верхню і нижню межі передбачуваного прибутку за надання сервісу $S_i$ у визначений період за відомими наведеними у додатку В формулами (В.3), (В.4) і (В.5) відповідно.

На кроці 6 повторюємо обчислення тих же показників, але вже для оцінювання кожного пакету сервісів, які надаються провайдерам.

Результати обчислення зазначених показників для пакетів сервісів надаються ОПР для прийняття остаточного рішення на кроці 8.

Результати виконаного експериментального дослідження ефективності застосування статистичного методу групування пакетів сервісів на даних ІТ-компаній і провайдерів наведені у додатку В. Детальніше розглянемо формування вхідних даних з врахуванням специфіки галузі інфокомунікацій. Ілюструватимемо використання статистичного методу на прикладі пакету сервісів, який містить сервіси $S_1$,$S_2$,$S_3$. Статистика замовлень цих сервісів представлена за певний період часу у таблиці В.1. Ціни надання сервісів

наведено у таблиці В.2. Витрати на підтримку сервісів за розглянутий період часу наведено у таблиці В.3.

Результати проведеного експериментального дослідження наведені у таблиці В.4. Тут значення передбачуваного прибутку провайдера за визначений період, ризику, верхньої і нижньої межі для згаданого вище пакету сервісів розраховані на основі відповідних значень сервісів пакету. Саме зазначені параметри для пакету сервісів у цілому використовує ОПР при прийнятті рішення щодо впровадження цього пакету сервісів.

## 3.2 Групування сервісів у пакети на основі алгоритмів кластеризації

Задача групування сервісів у пакети з використанням формальної постановки 2, наведеної у підрозділі 1.4, дозволяє використовувати успішний досвід формування пакетів адміністративних послуг на основі алгоритмів кластеризації [128]. Але кластеризацію будемо застосовувати до даних ІТ-компаній і провайдерів у галузі інфокомунікацій та їх клієнтів.

Ідея застосування алгоритмів кластеризації відповідає сутності групування сервісів у пакети для надання ІКП як ІТ-компаніями провайдерам, так і провайдерами своїм клієнтам. Дійсно, результати розв’язанні задачі кластеризації на основі даних накопичених в ІС провайдера умов про надання пакетів сервісів є дуже інформативними для ОПР. Кожний отриманий кластер можна розглядати як основи відповідного пакету сервісів, який можна пропонувати клієнтам. Подібним чином кластери, отримані на основі накопичених ІТ-компаніями даних можна пропонувати як відповідні пакети сервісів для провайдерів. Детальніше розглянемо основні кроки алгоритму.

Крок 1. Підготовка і введення вхідних даних. Алгоритм обробляє вхідні дані, подані у вигляді наведених у додатку В таблиць В.5 і В.6. Перша з таблиць прив’язує сервіси до угод, її рядки відповідають угодам, а стовпці – базовим сервісам. На основі даних цієї таблиці легко встановити для кожного сервісу $S_j$ угоди, згідно яких він надавався. Відповідно і для кожної угоди $U_i$ легко встановити ті сервіси, умови надання яких вона визначала. Про це

свідчать значення 1 або 0 відповідних рядку *i* і стовпці *j* (див. додаток В). Друга таблиця містить дані про кількості замовлень сервісів за визначені користувачем періоди часу.

Крок 2. Побудова кластерів за допомогою наведених у додатку В відомих алгоритмів. Важливою умовою вибору алгоритму є його здатність визначати кластери без обумовленої користувачем їх кількості. Це дозволяє надавати ОПР більш придатну для прийняття остаточного рішення щодо створення на основі побудованих алгоритмом кластерів пакетів сервісів. Визначені кластери підлягають оцінюванню для надання ОПР додаткової інформації щодо їх пріоретизації для прийняття остаточного рішення.

Крок 3. Для пріоретизації кластерів доцільно використовувати наведені у додатку В відомі показники, які свідчить про наявність чи відсутність прихованої закономірності, властивої сервісам одного кластера. Ці показники обчислюються на основі накопичених даних для кожної пари сервісів. Оскільки оцінювання базується на попарному порівнянні сервісів, важливо мати шкалу для ранжування ступеня впливу цих показників на групування сервісів у пакети. Використовуються відомі шкали, теж наведені у додатку В. Наявність згаданої вище закономірності, оцінки взаємної обумовленості сервісів дуже важливі для прийняття рішення щодо пакету сервісів.

Крок 4. Узагальнюємо результати оцінювання пар сервісів для кожного отриманого кластеру. Використовуємо показники, отримані на кроці 3, а також результати додаткових перевірок взаємозв’язків сервісів, які дозволяє легко інкорпорувати розглянутий підхід. Зокрема, важливе співпадіння часу надання і користувача сервісів пари.

Крок 5. Отримані оцінки разом з даними про вартість кожного сервісу та витрати на його надання передаються ОПР для визначення доцільность впровадження кластерів як пакетів. Кластер, значення показників взаємовпливу кожної пари сервісів якого є високим за прийнятою шкалою приймається як основа формування пакету інфокомунікаційних сервісів за умови відповідності економічних показників. Додаткове рішення ОПР

вимагається у випадку кластерів, показники взаємовпливу кожної пари сервісів яких не перевищують визначеного порогу за прийнятою шкалою, але мають сприятливі економічні показники.

Спроби використання для визначення доцільності групування сервісів кластера у пакет на основі інших статистичних показників, наприклад [131], вимагають додаткового експериментального дослідження.

### 3.3 Комбінований метод групування сервісів у пакети

Необхідність врахування специфіки задачі групування сервісів у пакети, встановлена у розділі 1, викликала потребу у трьох її формальних постановках. Обґрунтування для розв'язання цієї задачі статистичного методу та підходу на основі кластеризації стосується перших двох формальних постановок. Щоб реалізувати можливості, які надає формальна постановка 3, необхідно запропонувати відповідні методи, що враховують її особливості. Якщо, перші дві формальні постановки орієнтовані на використання накопиченого досвіду надання сервісів пакетами, то тут потрібні методи пошуку найкращих за формальними критеріями розв'язків з врахуванням системи обмежень. Виходом у цій ситуації може бути застосування методів математичного програмування і метаевристичихні алгоритмів.

Щоб зробити обгрунтований вибір детальніше проаналізуємо особливості наведеної у підрозділі 1.4 формальної постановки 3 задачі групування сервісів у пакети. Перша її важлива особливість, яка впливає на вибір методу, полягає у використанні двох типів змінних для визначення структури розв'язків. Дійсно, булеві змінні $v_{ij}$ визначають належність сервісу $S_i$ до пакету, який буде надаватися провайдеру $P_j$, а дійсні змінні $r_{ij}$ – знижки до преференційної ціни. Друга важлива особливість задачі, яка суттєво впливає на вибір методу, полягає у наявності окремих цільових функцій вигоди – для IT-компанії (1.1) і кожного провайдера (1.2). При цьому значення кожної з функції залежить від змінних обох визначених груп. Третьою важливою для вибору методу особливістю задачі є система різноманітних

обмежень (1.3)–(1.6) – лінійних і нелінійних. Вони стосуються знижок, наявних ресурсів, економічної доцільності, технологічної узгодженості сервісів, згрупованих у пакети.

Отже постає потреба у ефективних методах вирішення двокритеріальної задачі змішаного математичного програмування великих розмірів. Виконаний аналіз показав, що специфіка задачі вимагає розроблення нових і вдосконалення існуючих методів, які враховують специфічні особливості її формальної постановки. Їх порівняння дозволить вибирати найбільш придатний метод в кожній ситуації, враховуючи такі додаткові чинники як необхідні ресурси і час, відведений на отримання розв'язку задачі.

Перш ніж перейти до запропонованих методів розв'язання посталої задачі, розглянемо сучасні концепцій, використання яких це буде вимагати. Результати огляду літературних джерел підтверджують доцільність застосування декомпозиції, метаевристичного підходу і пошуку компромісних розв'язків. Розпочнемо з переваг застосування концепції декомпозиції.

У нашому випадку природним буде застосування декомпозиції сформульованої задачі на однокритеріальну підзадачу для IT-компанії та однокритеріальні підзадачі для кожного провайдера, щоб оцінити їх можливу економічну вигоду і потім шукати компроміс, беручи до уваги критерії вигоди кожної із сторін. Розглянемо зазначені вище однокритеріальні підзадачі.

Однокритеріальна підзадача для IT-компанії полягає у знаходження найкращого розв'язку за критерієм максимізації цільової функції (2.1) за умови дотримання обмежень усіх типів (1.3)–(1.6).

Однокритеріальна підзадача формулюється для кожного провайдера $P_j, j = 1, \dots, m$. Для провайдера $P_j$ вона полягає у знаходженні найкращого розв'язку для цього провайдера за критерієм максимізації цільової функції

$$Q_j = \sum_{i=1}^{m} (p_{ij} - d_{ij}(1 - r_{ij}))v_{ij}; \qquad (3.1)$$

за умови виконання обмежень (1.3)–(1.6).

Перейдемо до метаевристик. Запропоновані дві моделі підзадач відрізняються критеріями вигоди сторін – IT-компанії і провайдерів. Однак

вони мають багато спільних аспектів, зокрема систему обмежень. У цьому випадку свою доцільність демонструють метаевристики, дозволяючи взяти до уваги особливості підзадач і застосовуючи спільні ефективні алгоритми.

Насамкінець уточнимо роль компромісу. Результатом розв'язання кожної з наведених підзадач є пакет сервісів, який забезпечує максимальний дохід для відповідної сторони. На основі розв'язків підзадач для ІТ-компанії та провайдерів необхідно знайти компромісний розв'язок, який враховує наявні обмеження. Тут доцільним виглядає перебір взаємних поступок сторін з врахуванням знижок. Саме пошук компромісного розв'язку складе основу для вдосконалення об'єднаного розв'язку підзадач.

Для розв'язання задачі групування сервісів у пакети в роботі розроблений новий комбінований метод і удосконалений керований генетичний алгоритм. Перейдемо до описання першого з них.

Комбінований метод базується на використанні зазначених вище трьох концепцій. Для традиційної декомпозиції на підзадачі специфічним є виділення підзадач для ІТ-компанії і кожного провайдера $P_j$, $j = 1, \dots, m$. Для розв'язання кожної з зазначених підзадач вибрано унікальний набір сучасних методів і алгоритмів, модифікованих з врахуванням їх специфіки. Оригінальним аспектом застосування метаевристик на етапах знаходження розв'язків описаних вище підзадач є поєднання жадібного підходу з ймовірнісним. Цим закладено основу для врахування при розв'язанні задачі наближення до оптимуму і часових затрат. Особливістю застосування третьої концепції – пошуку компромісу є евристичний алгоритм, що уможливлює взаємні поступки ІТ-компанії та провайдерів при прийнятті рішень. По суті кожна концепція відповідає етапу розв'язання задачі групування сервісів у пакети – декомпозиція задачі на підзадачі (етап 1), розв'язання підзадач (етап 2), пошук компромісного розв'язку (етап 3),

Тепер зосередимося на описанні і обґрунтуванні алгоритмів і методів, які інтегруються запропонованим новим комбінованим методом. З метою досягнення високої точності розв'язків для кожного з етапів розроблено

декілька алгоритмів. Вибір найбільш придатних з них залежить від специфіки задачі, результатів виконаних експериментальних досліджень, врахування взаємної вигоди і обмежень на час розв'язання задачі.

### 3.3.1 Алгоритми комбінованого методу для розв'язання підзадач

Щоб сконцентруватися на основних аспектах задачі групування сервісів у пакети, рел'єфно виокремити переваги розроблених алгоритмів пошуку розв'язків доцільно дещо спростити математичну модель. Зазначимо, що спрощення не мають змінити сутність моделі. Будемо розуміти під сервісом групу взаємопов'язаних сервісів, а ресурсні обмеження зведемо до одного ресурсу IT-компанії. Тоді в обмеженнях (1.3) $L$ прийме значення 1 і параметр $\beta_{ijl}$ можна подавати у вигляді $\beta_{ij}$. Це не вплине на сутність моделі. Аналіз показав, що при групуванні сервісів у пакети найчастіше один ресурс – кадровий, інфраструктурний або часовий – є критичним і впливає на рішення щодо надання сервісів. Звичайно обсяги інших ресурсів перевищують потреби. Без втрати загальності моделі можемо прийняти, що обмеження (1.4)–(1.5) будуть задовольнятися при всіх допустимих розв'язках. Тепер сконцентруємося на детальному описанні реалізації основоположних концепцій комбінованого методу групування сервісів у пакети.

Реалізація етапу 1 призвела до декомпозиції задачі на описані вище однокритеріальну підзадача для IT-компанії та множину однокритеріальних підзадач для провайдерів.

Для реалізації етапу 2, пов'язаного з розв'язанням виділених підзадач, виходячи з характеру цільових функцій і обмежень, доцільно розробити спільні алгоритми. Закладена в основу прийнятого вище підходу до групування сервісів у пакети ідея врахування в процесі розв'язанні задачі наближення до оптимуму і часових затрат обумовлює використання, з однієї сторони, алгоритму, який поєдує ймовірнісний і жадібний вибір, з другої сторони, алгоритму мурашиної колонії. Перейдемо до їх описання.

*Ймовірнісно-жадібний алгоритм розв'язання підзадачі для ІТ-компанії*

Врахувати специфіку задачі дозволяє поєднання жадібного і ймовірнісного вибору. Виникає потреба у впровадженні спільного показника, який характеризує розв'язки з врахуванням сервісів, провайдерів, їх групування і знижок, ресурсу. З цією метою на першому кроці етапу будемо обчислювати для кожної пари «сервіс $S_i$ – провайдер $P_j$» цінність одиниці ресурсу. Цінність ресурсу будемо визначати часткою доходу від надання сервісу, яка припадає на цей ресурс, і обчислювати за допомогою формули:

$$\theta_{ij} := \frac{d_{ij}(1 - r_{ij})}{\beta_{ij}}. \quad (3.2)$$

Використовуючи обчислені цінності одиниці ресурсу $\theta_{ij}$ інтегруємо жадібний та ймовірнісний вибір при розв'язанні підзадачі для ІТ-компанії:

1) для реалізації базової ідеї жадібного вибору:

- упорядковуємо згадані вище пари «сервіс $S_i$ – провайдер $P_j$» по незростанню цінності одиниці ресурсу;
- вибираємо пари у визначеному порядку, групуючи сервіси кожного провайдера (чи вибраних провайдерів) у пакети, дотримуючись виконання ресурсного обмеження;

2) для реалізації базової ідеї ймовірнісного вибору:

- з заданою періодичністю запускаємо процедуру ймовірнісного вибір пари «сервіс $S_i$ – провайдер $P_j$»;
- обчислюємо оцінки імовірності і виконуємо вибір пар на їх основі.

Детальна схема ймовірнісно-жадібного алгоритму розв'язання на етапі 2 підзадачі для ІТ-компанії наведена у додатку Б. Більш детально опишемо особливості виконання найважливіших кроків цього алгоритму.

По-перше, жадібний вибір спрямований на швидке досягнення найкращого розв'язку. Втілення класичного жадібного критерію «максимум вигоди на одиницю витрат» базується на пріоритетності призначень – ефективніші призначення сервісу $S_i$ провайдеру $P_j$, тобто призначення чиї значення $\theta_{ij}$ більші, вибираються першими.

По-друге, жадібному вибору притаманні пастки локальних максимумів. Саме їх уникненню сприяє ймовірнісний вибір. Він дозволяє розширити простір розв'язків, додаючи перспективні пари «сервіс $S_i$ – провайдер $P_j$». При цьому ймовірності вибору пар пропорційні їх цінності $\theta_{ij}$.

По-третє, ймовірнісно-жадібному алгоритму, який ефективно поєднує пріоритетне призначення пар $S_i - P_j$ з випадковістю їх вибору, пртаманні інші недоліки. Йому бракує навчання на основі накопиченого досвіду з використанням довгострокових стратегій неперервного поліпшення поточних розв'язків. Цей недолік, який пояснюється відсутністю колективної пам'яті, а також недостатня масштабованість, дисбаланс дослідження і використання вимагають використання ефективних алгоритмів, позбавлених зазначених недоліків. До таких належить алгоритм мурашиних колоній.

*Алгоритм мурашиних колоній розв'язання підзадачі для ІТ-компанії*

Схема алгоритму мурашиних колоній для розв'язання підзадачі для ІТ-компанії наведена у додатку Б. Детальніше розкриємо особливості реалізації його найважливіших кроків. По-перше, для позбавлення від недоліків ймовірнісно-жадібного алгоритму тут поєднуються локальна евристика (цінність пари) з феромонною пам'яттю, що підтримує самоорганізоване й адаптивне навчання. Таким чином уможливлюється гнучке балансування між дослідженням нових комбінацій і поліпшенням поточних найкращих розв'язків. Тому цей алгоритм виявився дуже ефективним для розв'язання підзадач ІТ-компанії і провайдерів в рамках розв'язання задачі групування сервісів у пакети в ІС провайдерів ІКП.

По-друге, вимагають пояснення необхідні для описання алгоритму параметри. Тут початковий рівень феромону $\tau_0$ визначає початкове значення кількості феромону для кожної можливої пари $S_i - P_j$. Вага феромону $\alpha$ визначає ступінь впливу феромонної інформації на прийняття поточного рішення мурахою. Загалом прийнято, чим більше значення $\alpha$, тим більший вплив має накопичений досвід. Вага евристичної інформації $\beta$ визначає, наскільки мураха орієнтується на локальну цінність призначення $\theta_{ij}$.

Традиційно, велике значення ваги β посилює жадібну поведінку, тобто локальну ефективність. Коефіцієнт випаровування феромону $\rho \in (0,1]$ визначає швидкість втрати феромону з часом. Він дозволяє контролювати забування старої інформації і сприяє гнучкості пошуку. Кількість мурах в популяції, яка використовується в обчисленнях, позначається $A$.

По-третє, детальніше опишемо сутність визначальних кроків алгоритму, а саме – обчислення ймовірностей і верхньої межі значень цільової функції, оновлення феромонів та умову завершення роботи алгоритму. Вони визначають сутність алгоритму і забезпечують його ефективність. Найважливішим є обчислення ймовірності переходу для побудови розв'язку, під якою ми розуміємо ймовірність вибору мурахою наступної пари «$S_i - P_j$»:

$$p_{ij} := \frac{(\tau_{ij})^{\alpha}(\theta_{ij})^{\beta}}{\sum_{(i'j')_{ij} \in E}(\tau_{i'j'})^{\alpha}(\theta_{i'j'})^{\beta}}, \tag{3.3}$$

де $\tau_{ij}$ – поточний рівень феромону для пари $S_i - P_j$; $\theta_{ij}$ – цінність призначення $S_i - P_j$; $E$ – множина допустимих призначень (тих призначень, для яких на поточному кроці вистачає ресурсу).

На другому кроці необхідно обчислити верхню межу значень цільової функції. Для цього будемо використовувати правило вибору найкращих за ефективністю пар до вичерпання ресурсу. Для реалізації правила упорядкуємо всі пари $S_i - P_j$ за незростанням величини $\theta_{ij}$, а потім, доки вистачає ресурсу $T$, будемо додавати відповідні парам доданки до суми, визначеної формулою:

$$W^{max} = \sum_{(i,j) \in F} (1 - r_{ij}) d_{ij}, \tag{3.4}$$

де $F$ є множиною перших пар в упорядкованому їх списку, для яких вистачає ресурсу, тобто виконується обмеження (1.3) трансформоване під один ресурс:

$$\sum_{j=1}^{k} \sum_{i=1}^{m} \beta_{ij} v_{ij} \leq T.$$

На третьому кроці оновлюються феромони. Це відбувається після кожної ітерації на кожному ребрі за допомогою формули:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \sum_{l=1}^{L} \Delta\tau_{ij}^{l}, \quad (3.5)$$

де $\Delta\tau_{ij}^{l}$ – об'єм феромону, наданий мурахою $l$ призначенню $S_i - P_j$:

$$\Delta\tau_{ij}^{l} = \begin{cases} \frac{W_l}{W_{max}}, \text{якщо призначення } S_i - P_j - \text{ в пакеті,} \\ 0, \text{інакше,} \end{cases} \quad (3.6)$$

де $W_l$ – значення цільової функції для розв'язку мурахи $l$; $W^{max}$ – верхня межа значень для цієї цільової функції для розв'язку мурахи.

Насамкінець визначимо умова завершення роботи алгоритму мурашиних колоній. Умовою завершення роботи алгоритму є досягнення фіксованої кількості ітерацій $N$. У дисертаційній роботі це значення, яке загалом залежить від розмірності задачі, встановлено експериментально.

*Евристичні алгоритми розв'язання підзадачі для провайдерів*

Для цієї підзадачі теж розроблено два алгоритми – ймовірнісно-жадібний та мурашиної колонії. Вони подібні описаним вище для розв'язання підзадачі для ІТ-компанії, але виконуються з точки зору вигоди провайдерів. Для кожної пари $S_i - P_j$ теж обчислюється цінність одиниці ресурсу. Але тут використовується формула (3.7), згідно якої цінність одиниці ресурсу визначається як відношення доходу провайдера сервісу до обсягу ресурсу, необхідного ІТ-компанії для його підтримки:

$$\theta_{ij} := \frac{p_{ij} - d_{ij}(1 - r_{ij})}{\beta_{ij}}. \quad (3.7)$$

### 3.3.2 Алгоритм пошуку компромісного розв'язку

Розпочнемо з формальної постановки задачі етапу 3, пов'язаної з пошуком компромісу між розв'язками підзадач для ІТ-компанії і провайдерів.

Дано: $v^1 = \{v_{ij}^1\}$ і $v^2 = \{v_{ij}^2\}$ – отримані на етапі 2 розв'язки підзадач для ІТ-компанії і для провайдерів відповідно.

Знайти: компромісний розв'язок $v = \{v_{ij}\}$, який узгоджує інтереси ІТ-компанії і провайдерів і служить основою для групування сервісів у пакети.

Алгоритм пошуку компромісного розв'язку наведений у додатку Б.

Розглянемо його важливі аспекти. По-перше, в плані реалізації концепції взаємної вигоди користувачеві надається можливість визначати ваги $\omega_1, \omega_2 \in (0,1)$ критеріїв ІТ-компанії та провайдерів відповідно, $\omega_1 + \omega_2 = 1$.

По-друге, у компромісний розв'язок вносимо кожну пару сервіс – провайдер $(S_i - P_j)$ наявну у обох розв'язках, формуємо множину пар, що залишилися у розв'язках ІТ-компанії або провайдерів.

По-третє, для пар, що залишилися: обчислюємо оцінки значущості за зваженим критерієм компромісу; упорядковуємо за незростанням значущості; покроково до вичерпання ресурсу включаємо упорядковані пари до розв'язку.

ОПР надається можливість багаторазово виконувати алгоритм, встановлюючи різні ваги критеріїв до отримання бажаного результату.

### 3.3.3 Експериментальне дослідження комбінованого методу

Спочатку проаналізуємо алгоритми розв'язання підзадачі для ІТ-компанії. В описаних нижче експериментах приймалися такі параметри задач: $k = m = 10$, $d_{ij} \in [20; 100]$, $\beta_{ij} \in [10; 50]$, $T \in [0.2 \sum_i \sum_j \beta_{ij}; 0.8 \sum_i \sum_j \beta_{ij}]$.

*Експеримент 1.* Метою експерименту було дослідити вплив параметра β на ефективність роботи алгоритму мурашиної колонії. У експерименті були використані такі значення параметрів моделі: $A = 10, \alpha = 1,\ \rho = 0.2,\ \tau_0 = 1$. Було отримано по 5 розв'язків для кожного значення $\beta = \in \{0.25, 0.5, 0.75, 1, 2, 3, 4\}$ і обчислене середнє значення цільової функції. Графік залежності значення цільової функції від параметра β подано на рис. 3.1. Результати експерименту свідчать на користь дотримання значення $\beta \approx 2 \div 3$ при застосуванні алгоритму. Дійсно після досягнення значення 2 показника β результати стабілізуються. Якість розв'язків зростає не суттєво. Це підкреслює доцільність дотримання саме значення $\beta \approx 2 \div 3$. У такому випадку використання алгоритму мурашиної колонії, а власне і комбінованого методу, буде найбільш доцільним, адже додаткові спроби поліпшити результат приречені на невдачу.

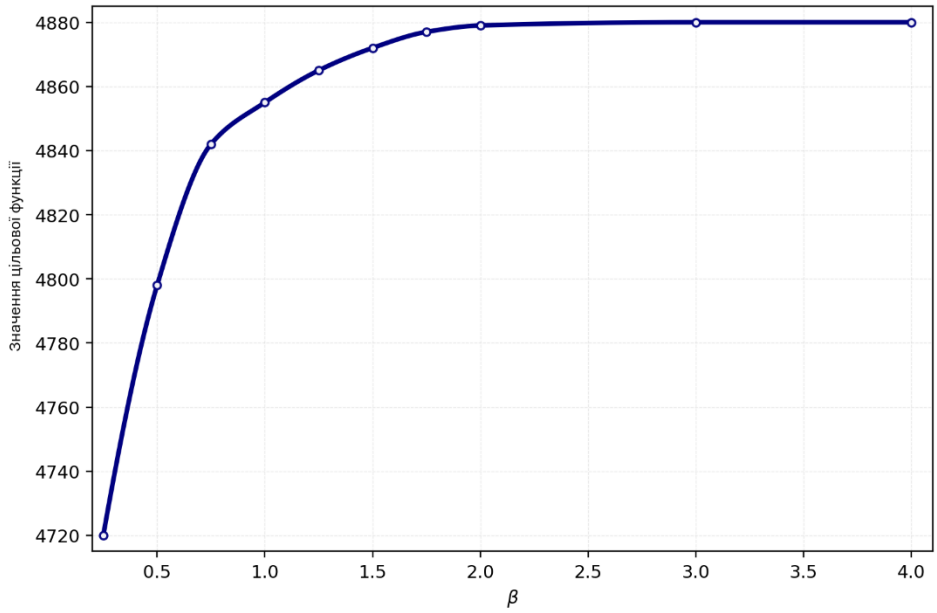


Рисунок 3.1 – Залежність значення цільової функції від значення β

*Експеримент 2.* Тут метою експерименту було дослідити вплив на збіжність алгоритмів параметра умови завершення $N$, який визначає кількість ітерацій для обох алгоритмів етапу 2. В експерименті для оцінки збіжності алгоритмів вони розпочинали виконуватися одночасно в однакових умовах. Наведені на рис. 3.2 результати експерименту подають динаміку значень цільової функції для отриманих найкращих на кожній ітерації розв'язків (рекордів) обох методів. Динаміка свідчить про перевагу алгоритму мурашиних колоній. Його результати, особливо на ранніх ітераціях, кращі.

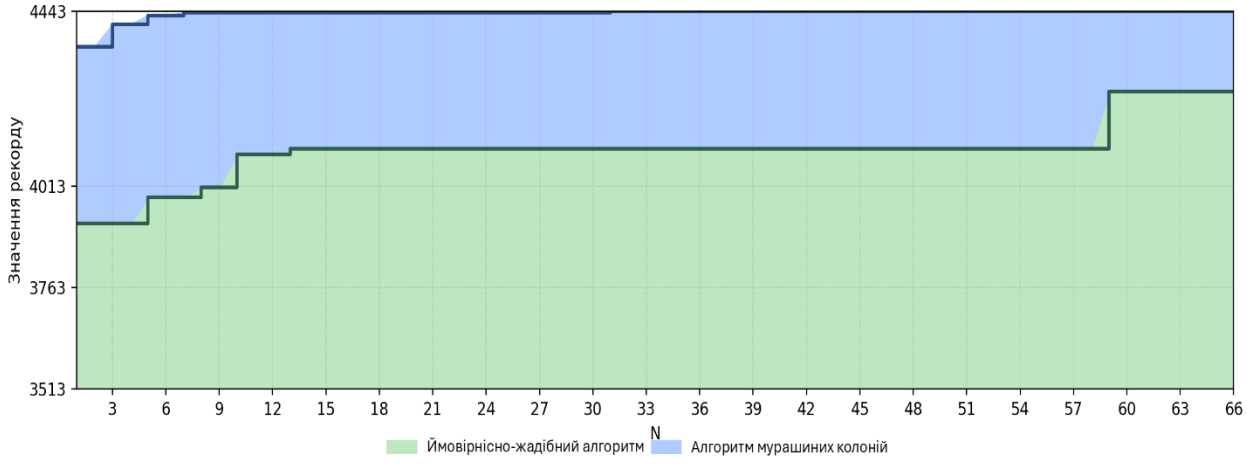


Рисунок 3.2 – Динаміка значень цільової функції по ітераціях

*Експеримент 3.* Оскільки мета полягала у встановленні рекомендованого значення параметра $N$, було аналізувалася його залежність від розмірності задачі. Було отримано розв'язки 5 задач, які генерувалося для кожного значення $N \in \{10, 100, 200, 300, 500, 1000, 2000, 3000\}$, встановлено середній час роботи алгоритмів та середні значення цільової функції. Результати експерименту зображені на рис. 3.3 і 3.4. Наведені на першому з них графіки відображають вплив параметра $N$ на середній час виконання кожного з алгоритмів, а наведені на другому – на середнє значення цільової функції .

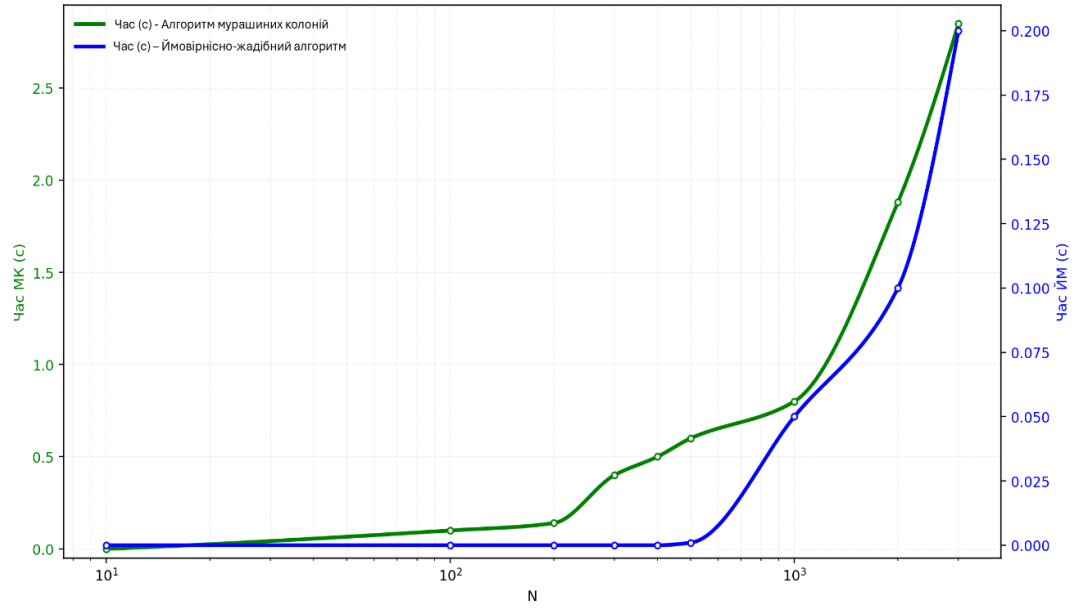


Рисунок 3.3 – Вплив кількості ітерацій $N$ на середній час роботи алгоритмів

Варто прийняти до уваги, що реалізація ймовірнісно-жадібного алгоритму базувалася на розпаралелювання обчислень. У такий спосіб вдалося значно зменшити час виконання алгоритму. Реалізація алгоритму мурашиних колоній передбачає розпаралювання лише дій окремих і виключно у процесі виконання окремої ітерації. А виконання ітерацій може бути лише послідовно, адже кожна з них завершується оновленням феромонного сліду. Отже переваги першого з алгоритмів у розпаралелюванні компенсують деякі його недоліки.

Зазначимо, що притаманне зазначеним алгоритмам поліпшення точності розв'язків із зростанням $N$, поступово слабшає внаслідок ефекту насичення.

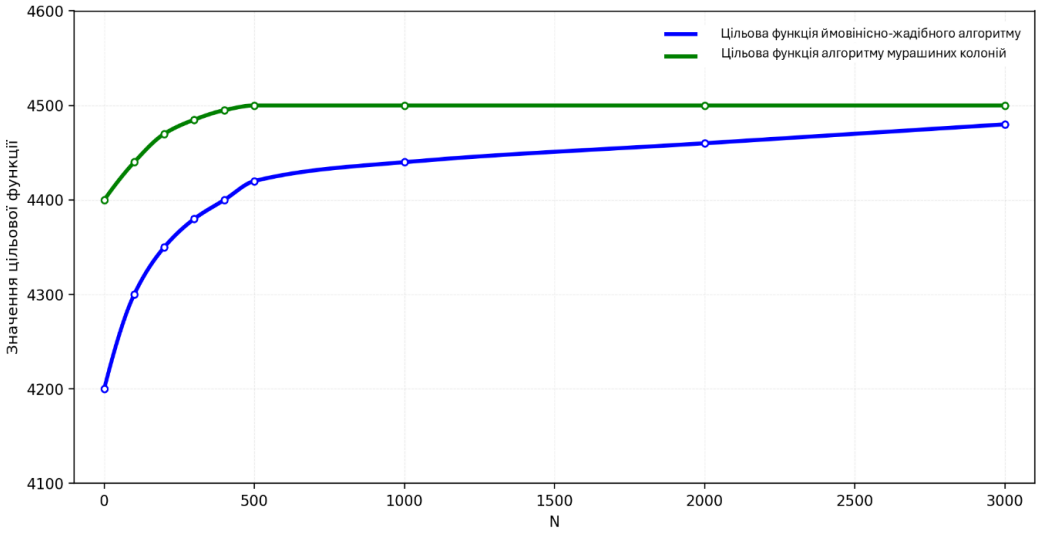


Рисунок 3.4 – Вплив кількості ітерацій $N$ на значення цільової функції

Найважливішим наслідком виконаних експериментів є встановлення найбільш придатного значення параметра $N$ з урахуванням розмірності задач групування сервісів у пакети. Для ймовірнісно-жадібного алгоритму доцільним є значення $N = 20\ km$, а для алгоритму мурашиної колонії – $N = 0.5\ km$. Більші значення позитивного впливу на точність, а витрати збільшать.

*Експеримент 4*. Метою цього експерименту було встановити вплив розмірів задачі на час роботи і точність кожного з зазначених алгоритмів. Для кожної розмірності $m \times k \in \{5 \times 5, 10 \times 10, 20 \times 20, 30 \times 30,\}$ було згенеровано і розв'язано 5 задач, результати яких представлені на рис. 3.5.

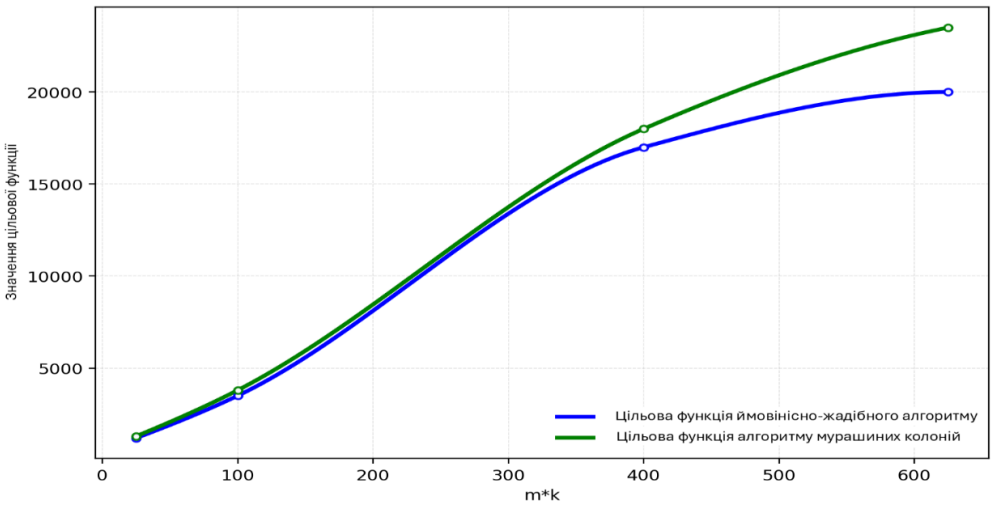


Рисунок 3.5 – Залежність значення цільової функції від розмірності задачі

Побудовані на основі даних, отриманих за допомогою кожного з алгоритмів, графіки залежності значень цільової функції від розмірності задачі дуже інформативні. Вони свідчать, що зростання розмірності задачі менш впливає на точність алгоритму мурашиної колонії. Його перевага за точністю очевидна. Отримані за його допомогою розв'язки мають до 12% вищі значення цільової функції, ніж розв'язки отримані за допомогою ймовірнісно-жадібного алгоритму. Графіки залежності часу роботи зазначених алгоритмів від розмірності задачі наведені на рис. 3.6.

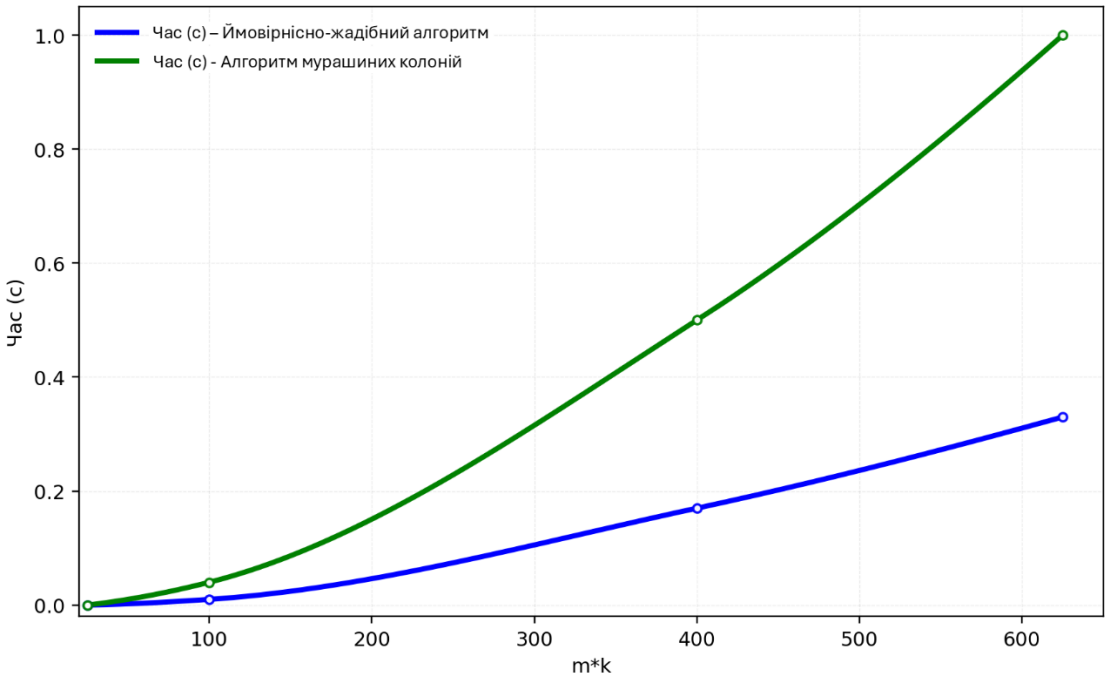


Рисунок 3.6 – Залежність часу виконання від розмірності задачі

Загалом виконане експериментальне дослідження показало переваги використання у комбінованому методі алгоритму мурашиної колонії над ймовірнісно-жадібним алгоритмом. По-перше, навіть при розв'язанні задач великої розмірності  час його роботи можна вважати прийнятним. По-друге, точність отриманих за його допомогою розв'язків значно перевищує точність розв'язків, отриманих за допомогою іншого алгоритму. Це підтверджено як при розв'язанні підзадачі IT-компанії, так і підзадач провайдерів.

Комплексна перевірка запропонованих у дисертаційній роботі комбінованого методу і удосконаленого керованого генетичного алгоритму

була виконана за допомогою розробленого у складі платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП компонента групування сервісів у пакети. Результати проведених експериментів підтвердили працездатність розроблених комбінованого методу і удосконаленого керованого генетичного алгоритму. Для експериментальної перевірки точності розроблених комбінованого методу і удосконаленого керованого генетичного алгоритму їх розв'язки порівнювалися з розв'язками, отриманими за допомогою методу повного перебору. Для оцінювання точності зазначених методу і алгоритму обчислювалося середнє відхилення суми значень цільових функцій ІТ-компанії та провайдерів, отриманих за їх допомогою з оптимальним значенням, отриманим повним переборомзнайденого повним перебором.

Результати експерименту наведені у таблиці 3.1. Висока обчислювальна складність методу повного перебору не дозволила виконати перевірку точності розв'язання задач групування сервісів у пакети великих розмірів. Але результати проведеного порівняльного аналізу на задачах невеликих розмірів є підставою для висновку, що точність комбінованого методу буде перевищувати точність модифікованого керованого генетичного алгоритму і на задачах великих розмірів.

Таблиця 3.1 – Середнє відхилення значення цільової функції від оптимуму (у %)

| Розміри задачі $m \times k$ | Точність комбіно-ваного методу | Точність модифікованого керованого генетичного алгоритму |
|---|---|---|
| $3 \times 3$ | 2,1 | 2,3 |
| $4 \times 4$ | 3,0 | 3,7 |
| $5 \times 5$ | 3,8 | 4,2 |
| $6 \times 6$ | 4,2 | 5,8 |
| $7 \times 7$ | 5,4 | 6,9 |
| $8 \times 8$ | 6,2 | 8,3 |

Підтверджена експериментально придатність комбінованого методу для розв'язання задачі групування сервісів у пакети у наведеній у підрозділі 1.4 формальній постановці 3 уможливлює створення і широке застосування відповідного компонента у складі платформи підтримки ЖЦ сервісів у ІС

провайдерів ЦКП. Це ще важливо і тому, що сприяє автоматизації одного з фундаментальних процесів ЖЦ сервісів. Крім достатньої точності отриманих розв'язків цей метод також відповідає нефункціональним вимогам до платформи. Окрім згаданої точності, можна говорити про ефективність, продуктивність, масштабованість,…. Як один із групи запропонованих для розв'язання задачі групування сервісів у пакети комбінований метод має також набір додаткових прихованих ознак, про які буде свідчити його використання в компоненті групування сервісів. Це робить виправданим розроблення для реалізації у зазначеному компоненті методу урахування опіній бізнес- і IT-менеджерів для вибору найбільш придатного у кожній ситуації методу.

Загалом експериментальне дослідження підтвердило практичну доцільність використання комбінованого методу для розв'язання задачі групування сервісів у пакети. Отримані результати вказують на перспективність його використання у компоненті групування сервісів у пакети у складі платформи підтримки ЖЦ сервісів для подальшого впровадження в діяльність провайдерів ІКП.

## 3.4 Варіант генетичного алгоритму групування сервісів у пакети

Для урахування особливостей розв'язуваної задачі був запропонований варіант керованого генетичного алгоритму на основі модифікації генетичного алгоритму Голанда [134]. Традиційні поняття генетичних алгоритмів включають створення популяції, відбір особин-батьків, реалізацію схрещення (кросинговеру), мутації, відновлення та оновлення популяції. Розглянемо особливості розроблених операторів варіанту керованого генетичного алгоритму.

### 3.4.1 Оператори варіанту керованого генетичного алгоритму

*Розпочнемо зі створення початкової популяції.* Початкова популяція містить задану кількість випадковим чином створених особин без повторів. Як прийнято, кожну особину будемо оцінювати мірою «пристосовуваності до умов існування». Під пристосованістю особини *y* розумітимемо сумарне

значення функцій доходів ІТ-компанії і всіх провайдерів на матриці $Y$ змінних розв'язку, який відповідає особині:

$$f(y) = F(y) + \sum_{j=1}^{m} \Phi_j(y).$$

*Вибір батьків* відбувається за правилом рулетки Гольдберга [135]: ймовірність вибору особини $y^i$ в якості батька пропорційна її пристосованості:

$$p(y^i) = \frac{f(y^i)}{\sum_{i=1}^{N} f(y^i)},$$

де $f(y^i)$ – пристосованість особини $y^i$, $N$ – кількість особин у популяції.

На кожному кроці за цим правилом обирається $\frac{N}{4}$ пар особин.

*Перейдемо до схрещення.* Для пари відібраних особин-батьків виконується одноточковий кросинговер. Для цього випадковим чином обирається точка розриву і утворюються два генотипи нащадків.

*Мутація*

Мутація особини здійснюються за імовірнісною схемою шляхом зміни кожного біта на протилежний з ймовірністю 1/10000. Оскільки матриця $Y$ має розмірність $q \times m$, то ймовірність мутації особини становить $(q \times m)/10000$ і за невеликих значень добутку $q \times m$ мутація особини може не відбутись. У цьому випадку спроба мутувати цю особину повторюється.

*Контроль збіжності алгоритму*

Для контролю наближення результату до найкращого розв'язку розроблена система правил на основі принципу переключення режимів – пошук прискорюється, якщо в процесі роботи алгоритму кроки наближення досить великі, алгоритм переключається на розширення використання корисних хромосом, якщо кроки наближення зменшуються.

Система правил (що описана нижче) підтримує баланс між розширенням простору пошуку оптимуму і ефективним використанням усіх корисних для одержання найкращого розв'язку хромосом.

При цьому отримання нової популяції особин на базі представників минулої епохи виконується не за рахунок послідовного виконання операторів кросинговеру та мутації, а забезпечується балансування між пошуком:

– нових перспективних для отримання ефективних розв'язків областей, які можуть містити шуканий оптимум, що не входять до вивченого простору мутацій кращих представників поточної популяції (за допомогою мутації);

– найкращих розв'язків, які можуть бути як локальними так і глобальними, у нових областях (за допомогою кросинговеру).

Який оператор (кросинговеру чи мутації) буде використано для отримання популяції наступної епохи визначається на основі правил 2 та 3. Алгоритм зосереджується на кросинговері доки поточний розв'язок суттєво поліпшується. Коли ж зростання локального оптимуму сповільнюється, відбувається перехід до пошуку нових областей з кращими розв'язками.

*Відновлення*

В процесі схрещення та мутації можуть бути отримані недопустимі розв'язки. Для того, щоб забезпечити роботу тільки з допустимими за обмеженням (7) розв'язками, розроблено оператор відновлення (repair) особини, який базується на правилі «видали найгірший».

*Визначення рекордного розв'язку*

Оскільки маємо багатокритеріальну задачу, то для визначення рекордного розв'язку визначені правила переваг розв'язку над іншими за значеннями критеріїв за умови виконання обмежень. Концепція визначення рекордного розв'язку використовує наявність двох цільових функцій – для IT-компанії ($F$) і провайдерів ($\Phi = \sum_{j=1}^{m} \Phi_j$), беручи до уваги можливість знижок. Вимагається, щоб у процесі вирішення задачі значення обох функцій покращувалися або погіршення значення однієї функції компенсувалося значнішим поліпшенням іншої (правило 5).

*Оновлення популяції*

Особина додається до популяції за умови, що вона є більш пристосованою, ніж найгірша особина, і (для уникнення повторів) не міститься

в поточній популяції. Узагальнена схема модифікованого варіанту керованого генетичного алгоритму наведена у додатку Б.

Кроки 8-19 реалізують модифікований керований генетичний алгоритм (КГА) розв'язання задачі $\boldsymbol{T}(w)$ булевого програмування для конкретного значення матриці знижок $w$. *Умова 1* завершення КГА задана правилом 1.

Зовнішній цикл while 7-23 алгоритму керує значеннями знижок $w$ відповідно певної стратегії. *Умова 2* завершення керованого алгоритму задана правилом 6.

В якості стратегії встановлення знижок, пропонується взяти узагальнені певним чином стратегії, описані в підрозділі 6.1.1. Так, наприклад, за стратегією а) найбільш уживані сервіси можна визначати:

– по поточному рекордному розв'язку;

– по топ-розв'язках поточної популяції.

### 3.4.2 Описання розробленої системи правил

У модифікованому варіанті керованого генетичного алгоритму використовується відома ідея управління збіжністю алгоритму і впроваджена нова концепція відбору розв'язків з врахуванням багатокритеріальності задачі.

*Управління збіжністю алгоритму* націлене на досягнення обґрунтованого компромісу між кількістю кроків алгоритму і точністю розв'язків. Для цього застосовується система правил управління операторами вибору, кросинговеру і мутації в процесі розв'язання задачі на основі поточних результатів, історії і значень ряду додаткових параметрів [136]. Нові додаткові параметри враховують особливості задачі і надають генетичному алгоритмові здатності швидкого сходження до досить близького до оптимуму результату

Кроки управління збіжністю будуть важливі і для модифікованого керованого генетичного алгоритму в цілому. Для формулювання модифікованого варіанту керованого генетичного алгоритму необхідно змінити систему параметрів.

При цьому доцільно залишити традиційний коефіцієнт приросту популяції $k_{\text{п.п}} = \frac{l}{N}$, де $l$ – кількість нащадків, отриманих на поточній ітерації генетичного алгоритму, $N$ – розмір популяції, і ввести додаткові параметри:

1) Для оцінювання ступеня наближення поточної популяції до оптимуму замість традиційного тиску відбору $\rho = \frac{f_{max}}{\bar{f}}$, де $f$ – цільова функція пошуку, $f_{max}$ – максимальне значення цільової функцій для поточної популяції, а $\bar{f}$ – середнє значення цільової функції поточної популяції, введемо *двосторонній тиск відбору*, як величину відношення максимальної суми значень обох цільових функцій до середньої суми значень обох цільових функцій популяції поточної епохи. При цьому приймемо, що критерієм оптимальності є максимізація значення функції пристосованості. Тоді двосторонній тиск відбору $\rho$ дорівнює визначається як $\rho = \frac{F_{max}+\Phi_{max}}{\bar{F}+\bar{\Phi}}$, де $F_{max}$ $(\Phi_{max})$ – максимальне значення цільової функції $F$ $(\Phi)$ для поточної популяції, $\bar{F}$ $(\bar{\Phi})$ – значення цільової функції IT-компанії (провайдерів) для поточної популяції (за визначенням традиційний тиск відбору і запропонований двосторонній тиск відбору не можуть бути меншими від одиниці);

2) Для оцінювання тенденції зміни збіжності генетичного алгоритму при переході від епохи до епохи замість швидкості зміни тиску $\Delta_{i-1,i} = \rho_{i-1} - \rho_i$, де $\rho_{i-1}, \rho_i$ – двосторонній тиск відбору, розрахований за результатами ітерацій $i-1$ та $i$ відповідно, будемо використовувати *відносну швидкість наближення до оптимуму*, як зваженої різниці між значеннями двосторонніх тисків відбору популяцій попередньої та поточної епох:

$$\widehat{\Delta}_{i-1,i} = \frac{\rho_{i-1} - \rho_i}{\rho_i}.$$

На їх основі розширимо набір правил, які в процесі роботи алгоритму будуть визначати оператори отримання популяції наступної епохи.

*Правило* 1. Можливі два традиційних варіанти завершення алгоритму:

– якщо $k_{\text{пп}} > k_{\text{гр}}$, де $k_{\text{гр}}$ – гранична (максимальна) можлива кількість згенерованих особин;

– якщо кількість послідовних ітерацій, впродовж яких не змінюється рекордний розв’язок, досягає встановленого граничного значення.

*Правило* 2. Правило формування популяції наступної епохи за допомогою оператора кросинговеру при знаходженні кращого розв’язку задачі з використанням введених параметрів набуває вигляду

$$\textbf{if } \widehat{\Delta}_{i-1,i} > 0 \textbf{ then } p = 1.$$

Логіка використання правила не змінилася – варто зберегти кращий розв’язок шляхом зміни стратегії «дослідження» на «використання».

*Правило* 3. Правило формування популяції наступної епохи за допомогою оператора кросинговеру в умовах невизначеності з використанням введених параметрів набуває вигляду

$$\textbf{if } \left(\left(\widehat{\Delta}_{i-1,i} \geq \widehat{\Delta}_{\text{гр}}\right)\textbf{and } \left(\rho_i \geq \rho_{\text{гр}}\right)\right) \textbf{ then } p = 1,$$

де $\widehat{\Delta}_{\text{гр}}$ – граничне значення відносної швидкості наближення до оптимуму, $\rho_{\text{гр}}$ – граничне значення двостороннього тиску відбору.

Правило реалізує традиційну логіку в умовах, коли характер збіжності процесів ще невизначений, бракує сходження до «локального» оптимуму.

*Правило* 4. Правило формування популяції наступної епохи за допомогою оператора мутації в умовах відсутності реального прогресу у пошуку кращого розв’язку задачі з використанням введених параметрів набуває вигляду

$$\textbf{if } \left(\left(\widehat{\Delta}_{i-1,i} < \widehat{\Delta}_{\text{гр}}\right)\textbf{or } \left(\rho_i < \rho_{\text{гр}}\right)\right) \textbf{ then } p = 0.$$

Це правило реалізує традиційну логіку – при суттєвому сповільненні процесу покращення розв’язку для отримання популяції наступної епохи використовується мутація, тобто здійснюється перехід від стратегії «використання» до стратегії «дослідження».

*Правило* 5. Додається правило визначення поточного рекордного розв’язку. При цьому не вимагається покращення рішення одночасно для ІТ-компанії (функція $F$) і для провайдерів (функція Ф). Логіка цього правила полягає в тому, щоб у процесі вирішення задачі використовувати покращення

суми значень обох функцій, коли погіршення значення однієї функції компенсувалося значнішим поліпшенням іншої.

*Правило* 6. Зберігається традиційне для керованого генетичного алгоритму правило зупинки роботи алгоритму і визначення розв'язку. Але умову закінчення змінимо – відносна швидкість повинна бути меншою від заданої впродовж певної кількості ітерацій.

Логіка традиційна – якщо функція пристосованості найкращої особини поточної погіршується відносно попередньої епохи, то перед зупинкою процес варто зробити кілька спроб пошуку потенційно ефективної області розв'язків, а вже у випадку невдачі зафіксувати правильну епоху з її результатом пошуку.

Для організації управління процесами у модифікованому варіанті керованого генетичного алгоритму користувач має задати початкові значення традиційних і нових параметрів $k_{\text{гр}}$, $\widehat{\Delta}_{\text{гр}}$, $\rho_{\text{гр}}$. До вибору і модифікації значень цих коефіцієнтів можна застосовувати традиційні для КГА рекомендації.

### 3.4.3 Дослідження варіанту керованого генетичного алгоритму

Для порівняльного аналізу ефективності розробленого алгоритму був проведений експеримент. Його метою було порівняти ефективності модифікованого варіанту керованого генетичного алгоритму і двоетапного алгоритму ЕА [71] для розв'язання задач різних розмірів, в яких $\omega = 0$. Результати аналізу зазначених алгоритмів наведені в табл. В.5 додатку В.

Отримані «не гарні» результати КГА для задач типу 3 пояснюються тим, що в цих задачах у кожного розв'язку величина $\sum_{l=1}^{q} \Phi_l$ значно перевищує величину $F$, тому згідно принципу компромісного вибору рекордного розв'язку, розв'язок, отриманий ЕА (метою якого є максимізація саме функції $F$) зазвичай буде найкращим. Зовсім інша ситуація має місце тоді, коли значення величин $F$ та $\sum_{l=1}^{q} \Phi_l$ є близькими або відрізняються не сильно, що і демонструють результати для задач типу 4.

Підвищення кількості особин популяції КГА підвищує його ефективність, але приводить до значного зростання часу розв'язання задач.

Загальна тенденція, яку дозволяють побачити експериментальні результати говорить про те, що кращі результати переважно надає модифікований варіант КГА, але зі зростанням розмірності задачі застосування КГА вимагає все більшого часу. Це підказує напрям для подальших досліджень, спрямованих на визначення критичної розмірності задач, для яких застосування КГА є виправданим.

Інший напрям подальших досліджень пов'язаний з інтеграцією запропонованих алгоритмів, що дозволяє зробити їх багатоетапний характер. Наприклад, з урахуванням отриманих результатів можна значно підвищити ефективність розробленого модифікованого варіанту КГА за рахунок того, щоб один з розв'язків початкової популяції був отриманий за допомогою ЕА.

### 3.5 Урахування опіній бізнес- і ІТ-менеджерів

Сьогодні ведення бізнесу в галузі інфокомунікацій вимагає зміцнення ролі менеджерів бізнесових і ІТ-служб при прийнятті рішень. Змінність умов діяльності провайдера обумовлює потребу у врахуванні додаткових чинників впливу. Тут на допомогу менеджерам приходить інтелектуальний асистент, описаний у розділі 2. Але видається доцільним також надання зазначеним менеджерам можливості впливати на вибір технологій розв'язання задач підтримки окремих процесів ЖЦ сервісів. Це стосується також групування сервісів у пакети. Врахування опіній менеджерів, побудованих на власному досвіді, сприятиме уникненню проблем і зростанню ефективності діяльності.

Використання запропонованих у розділі 3 методів і алгоритмів дозволяє врахувати специфіку задачі групування сервісів у пакети. Побудовані на ефективних широко вживаних методах та підходах вони забезпечують обґрунтовані близькі до оптимальних рішення. Але поява нових чинників впливу, які ще не враховані в реалізованих моделях і методах, вимагає швидкого рішення. Це можна зробити, зокрема, шляхом реалізації впливу опіній бізнес- і ІТ-менеджерів на вибір згаданих вище методів і алгоритмів.

У таких умовах варто вважати розв'язки, отримані за допомогою

наведених вище методів і алгоритмів, попередніми і рішення щодо надання пакету сервісів залишити за ОПР. При цьому для ОПР буде корисно врахувати опінії бізнес- і ІТ-менеджерів, які мають відповідний досвід. Але це вимагає розроблення методу врахування опіній бізнес- і ІТ-менеджерів при виборі методу розв'язання задачі групування сервісів у пакети.

Розглянемо чинники впливу, які необхідно взяти до уваги при розробленні методу врахування опіній бізнес- і ІТ-менеджерів при виборі методу розв'язання задачі групування сервісів у пакети. Для цього виконаємо таку послідовність кроків: 1) класифікуємо сутності предметної області діяльності, пов'язаної з наданням сервісів у галузі інфокомунікацій; 2) за допомогою зазначених сутностей виділимо важливі ознаки прийняття рішень щодо групування сервісів у пакети на основі опіній бізнес- і ІТ-менеджерів; 3) за цими ознаками вибудуємо основи зазначених інструментів, за допомогою яких будуть опрацьовуватися опінії бізнес- і ІТ-менеджерів для отримання інтегральної оцінки як основи остаточного вибору рішення.

Розпочнемо з першого кроку. За сутності предметної області діяльності провайдерів інфокомунікацій, пов'язані з наданням сервісів, приймемо:

1) клас провайдера за прийнятою системою класифікації – вплив класу провайдера сервісів дуже важливий для формування опінії бізнес- і ІТ-менеджерів, а наявність кількох класифікацій ускладнює ситуацію;

2) кількість користувальницьких сервісів провайдера – вплив цього чинника на формування опінії бізнес- і ІТ-менеджерів не викликає сумнівів;

3) кількість клієнтів провайдера згідно типу і сегменту ринку – цей чинник важливий, а наявність кількох класів клієнтів ускладнює ситуацію;

4) змінність умов ведення бізнесу – вимагається модифікація існуючих інструментів прийняття рішень або навіть створення нових інструментів прийняття рішень, у тому числі щодо групування сервісів у пакети;

5) типи сервісів OSS (IPTV, VoIP, білінгу та ін.) і BSS (управління інформацією користувача, замовленнями сервісів та ін.) – вплив типів сервісів істотний, а поява нових та модифікація існуючих сервісів ускладнює ситуацію;

6) додаткові вимоги провайдера до сервісів – дані про сервіси третіх сторін, майстрів даних провайдера впливають на опінії менеджерів;

7) функціональні вимоги до сервісів – формують основу для точнішої опінії бізнес- і ІТ-менеджерів щодо пакету сервісів;

8) нефункціональні вимоги до сервісів провайдера – дозволяють бізнес- і ІТ-менеджерам уточнити опінії щодо пакету сервісів;

9) оцінки пакетів сервісів, отримані за допомогою реалізованих методів і алгоритмів формування пакетів сервісів – оцінки, отримані за допомогою реалізованих методів кількісно візуалізують переваги одних пакетів сервісів над іншими, уточнюючи опінії бізнес- і ІТ-менеджерів.

Перейдемо до кроку обґрунтування на основі описаних вище сутностей важливих ознак інструментів прийняття рішень щодо групування сервісів у пакети на основі опіній бізнес- і ІТ-менеджерів. Важливо врахувати:

1) технології ШІ для комунікації з виконавцями, розуміння їх проблем і використання інструментів платформи для вирішення цих проблем;

2) інтеграцію опіній бізнес- і ІТ-менеджерів у інструменти підтримки процесів ЖЦ сервісів зокрема при групуванні сервісів у пакети;

3) колективне обговорення проблем, формування спільних рішень з інтеграцією різних опіній для поліпшення показників діяльності провайдера;

4) збір, накопичення, оброблення інформації про опінії бізнес- і ІТ-менеджерів, оцінювання ефективності пропонованих ними рішень у процесі вирішення проблем у минулому з метою використання у майбутньому;

5) техніки відбору бізнес- і ІТ-менеджерів для напрацювання колективних опіній при вирішення проблем на основі накопиченої в системі інформації щодо їх компетенцій і ефективності пропонованих раніше рішень;

6) концепції Agile-методологій, спрямованих на створення умов для підтримки і взаємодопомоги в середовищі бізнес- і ІТ-менеджерів в процесі обговорення проблем і напрацювання ефективних рішень;

7) активне впровадження технологій раннього виявлення проблем на основі накопиченого комплексу ознак і колективного обговорення їх результатів;

8) техніки формування команд виконавців для колективного обговорення проблем у визначених доменах бізнесової діяльності, що уможливлюють капіталізацію набутого досвіду;

9) відбір бізнес- і ІТ-менеджерів обмежити підрозділами, до сфери компетенцій яких належить розв’язання зазначеної задачі;

10) опрацювання зовнішніх джерел в термінах проблем, рішень, їх ефективності, обмежень, ризиків, актуальних опіній;

11) використання інструментів, побудованих на адекватних математичних моделях і методах оптимізації з врахуванням сучасних концепцій бізнесової діяльності.

Обґрунтувавши потребу застосування колективного опрацювання опіній для розв’язання задач підтримки процесів ЖЦ сервісів загалом і зокрема задачі групування сервісів у пакети, перейдемо до третього кроку. Необхідно конкретизувати описані вище на рівні ознак інструменти колективного опрацювання опіній. Зробимо це на прикладі команди бізнес- і ІТ-менеджерів провайдера інфокомунікацій, діяльність яких пов’язана з групуванням сервісів у пакети. Вибудуємо бачення інструментів опрацювання опіній бізнес- і ІТ-менеджерів для вибору остаточного рішення, спрямованого на взаємовигідний для провайдера і його клієнтів результат. Спрямований на досягнення бізнес-результатів в термінах вигоди і ресурсів, інструмент має враховувати сучасні концепції ведення бізнесу, зокрема змінність умов діяльності, врахування опіній бізнес- і ІТ-менеджерів та інших чинників.

По суті основна функціональність інструменту вже визначена. Важливо також надати для опрацювання опіній бізнес- і ІТ-менеджерів найбільш ефективні методи і алгоритми групування сервісів у пакети, зокрема розглянуті в роботі статистичний і комбінований методи, алгоритми кластеризації і модифікований керований генетичний алгоритм. Також важливо реалізувати функції організації колективного опрацювання опіній бізнес- і ІТ-менеджерами, зокрема вибору бізнес- і ІТ-менеджерів, введення

опіній, надання бізнес- і ІТ-менеджерам адекватної інформації щодо конкретної ситуації, в якій формується пакет сервісів, та ін.

У плані нефункціональних вимог колективне опрацювання опіній бізнес- і ІТ-менеджерів вимагає забезпечення певного рівня:

1) продуктивності (колективне опрацювання опіній обмежене часовими рамками, необхідний одночасний доступ багатьох користувачів);
2) надійності (необхідно зберігати значні обсяги накопичених даних);
3) масштабованості, високої доступності та зручності використання;
4) здатності до розвитку і обґрунтованості рішень та ін.

Цим вимогам відповідає метод, запропонований у праці [72]. Цей метод буде використаний при реалізації компонента групування сервісів у пакети в рамках платформи підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП.

**Висновки до розділу 3**

Розділ присвячений виконанню двох важливих завдань дисертаційного дослідження. Перше з завдань полягало у розробленні комплексу моделей і методів групування сервісів у пакети в ІС провайдерів ІКП на основі задачі змішаного програмування з використанням статистичних методів, алгоритмів кластеризації та еволюційних методів оптимізації.

У процесі виконання першого завдання обґрунтовано необхідність застосування широкого кола математичних моделей групування сервісів у пакети в ІС провайдерів ІКП і відповідних методів і алгоритмів, щоб взяти до уваги усі специфічні умови галузі. Встановлена доцільність використання статистичного методу, методу на основі алгоритмів кластеризації, варіанту комбінованого методу розв’язання змішаних задач математичного програмування і варіанту керованого генетичного алгоритму, а також методу урахування опіній бізнес- і ІТ-менеджерів для вибору найбільш придатного із вказаних методів і алгоритмів.

Застосування статистичного методу дозволило використати накопичений ІТ-компаніями і провайдерами ІКП досвід надання сервісів за

минулі періоди. Застосування методу для групування сервісів у пакети на основі алгоритмів кластеризації дозволило використовувати дані накопичених угод між IT-компаніями і провайдерами (між провайдерами і їх клієнтами).

Друге завдання полягало в удосконаленні комбінованого методу і керованого генетичного алгоритму багатокритеріальної оптимізації портфеля сервісів для досягнення взаємної вигоди IT-компанії, провайдерів з урахуванням SLA, ресурсних обмежень, міжсервісних залежностей і знижок.

Удосконалено комбінований метод групування сервісів у пакети, який відрізняється поєднанням декомпозиції, ймовірнісного і жадібного підходів та евристичного пошуку компромісу, що дозволили отримувати ближчі до оптимальних розв’язки з врахування системи знижок, узгоджувати точність і обчислювальні витрати та враховувати ринкові умови, що важливо для конкурентоспроможності IT-компаній і провайдерів.

Розроблено варіант керованого генетичного алгоритму для групування сервісів у пакети, який відрізняється додатковими параметрами і правилом знижок, що забезпечує досягнення глобального оптимуму у процесі вирішення задачі, коли значення обох функцій покращуються або погіршення значення однієї функції компенсується значнішим поліпшенням іншої, що дозволило отримувати близькі до оптимальних рішення за наявності двох цільових функцій − для IT-компанії і провайдерів − за умови виконання обмежень на ресурси, цінність сервісів, показники SLA та міжсервісні залежності з врахуванням системи знижок до преференційної ціни і надало можливість вибирати метод з врахуванням часу, відведеного на прийняття рішення.

Експериментально встановлено, що варіант керованого генетичного алгоритму та алгоритм мурашиних колоній дозволяють отримувати ефективні рішення в ситуаціях, яким притаманні невизначеність та ресурсні обмеження. Продемонструвало, що запропонований комплекс методів і алгоритмів дозволяє формувати пакети сервісів в різноманітних ситуаціях, що відрізняються критеріями, обмеженнями, додатковими чинниками впливу.

# 4 РОЗРОБЛЕННЯ ПЛАТФОРМИ ПІДТРИМКИ ЖИТТЄВОГО ЦИКЛУ СЕРВІСІВ

Розділ присвячено створенню і експериментальному дослідженню прототипу платформи підтримки ЖЦ сервісів в ІС провайдерів інфокомунікацій. На основі запропонованих у попередніх розділах моделей і методів розробляються інтелектуальний асистент, компоненти побудови архітектури сервісів і групування сервісів у пакети.

Розглядається реалізація запропонованих у попередніх розділах LLM, RAG-системи і нейронної мережі з глибоким навчанням для побудови архітектури сервісів, створення і експериментальне дослідження компоненту групування сервісів у пакети. Платформа підтримки ЖЦ сервісів містить компоненти для основних процесів і засоби природномовної комунікації з користувачами в контексті розуміння проблем, які вони розв’язують.

## 4.1 Аспекти реалізації компонентів LLM і RAG інтелектуального асистента

Для швидкої перевірки працездатності концепції інтелектуального асистента, як згадувалося вище, опишемо варіант реалізації взаємодії його важливих інтелектуальних компонентів LLM [142] і RAG [143]. Оскільки для спрощення вибрана робота з даними у форматі CSV, необхідно продемонструвати дві ключові здатності RAG-системи. Перша з них пов’язана з завантаженням таблиці у форматі CSV (у нашому прикладі з назвами пристроїв IBM, їх характеристиками). Друга полягає у наданні LLM доступу до цих табличних даних, щоб швидко генерувати релевантні відповіді на специфічні запитання користувача.

### 13.1.1. Загальний опис запропонованого рішення.

Запропонований варіант реалізації взаємодії компонентів LLM і RAG базується на очевидних логічно пов’язаних процесах перетворення даних CSV

у текстову базу знань, семантичного пошуку для отримання релевантного контексту, надання зазначеного контексту LLM для генерації відповіді.

З врахуванням особливостей реалізації зазначених процесів в умовах застосування вибраних інструментів реалізації основні кроки роботи системи можна представити у такий спосіб:

1. *Підготовка та завантаження даних у форматі CSV* – вміст CSV-файлу імпортується в структуру даних (DataFrame) з використанням бібліотеки Pandas. При цьому кожен рядок таблиці перетворюється у текстовий фрагмент, що містить пари «назва колонки: значення». Включення заголовків колонок до тексту є важливим для збереження контексту даних. Такий підхід – конвертувати кожен рядок CSV у текстовий «документ» – є прямим і ефективним для відносно невеликих таблиць з чітким семантичним змістом. У результаті формується база знань – список текстових записів, кожен з яких відповідає одному рядку CSV. Це значно скоротило час для перевірки працездатності концепції.

2. *Обчислення семантичних векторних представлень* – для кожного текстового запису бази знань генерується векторне представлення (embedding), що відображає семантику цього тексту. Для пришвидшення перевірки концепції двигуна у варіанті реалізації використано попередньо натреновану модель Sentence Transformers під назвою all-MiniLM-L6-v2, яка перетворює речення і абзаци у 384-вимірні вектори. Цей векторний простір сконструйований так, що семантично подібні тексти відповідають близьким векторним точкам. Векторні представлення всіх записів зберігаються у базі знань для подальшого пошуку.

3. *Семантичний пошук релевантного контексту* – коли користувач формулює запитання, його текст також перетворюється на вектор за допомогою тієї ж моделі embeddings. Після цього виконується обчислення косинусної міри подібності між вектором запиту та векторами всіх записів бази знань. Косинусна подібність характеризує схожість двох векторів як косинус кута між ними (значення в діапазоні від -1 до 1) і широко використовується для

порівняння текстових представлень, оскільки фокусується на напрямку векторів (семантичному змісті), а не на їхній довжині. Цей підхід дає змогу ранжувати документи бази знань за релевантністю до запиту користувача. Система відбирає топ-кілька записів із найбільшою косинусною схожістю (наприклад, 3 найрелевантніших фрагменти) як контекст для відповіді. Використання саме косинусної метрики забезпечує надійне визначення семантичної близькості запиту і текстів бази знань, завдяки чому бот фокусується лише на найбільш дотичних до питання даних.

4. *Формування підказки з контекстом для LLM* – вибрані на попередньому кроці текстові фрагменти (контекст) об'єднуються у підказку (prompt), яка буде передана LLM. Підказка починається зі службової інструкції, наприклад: “Базуючись на наступній інформації з CSV-даних: ...”, після чого наводяться знайдені фрагменти даних. Далі додається текст запитання користувача. Таким чином, LLM отримує всю необхідну фактичну інформацію перед формуванням відповіді. В підказці модель також явно просять давати відповідь лише на основі наданих даних та у разі відсутності потрібної інформації прямо вказувати на це. Такий метод забезпечує, що генерація відповіді відбувається з урахуванням релевантного контексту – модель фактично стає «обґрунтованішою» і менш схильною вигадувати факти.

5. *Виклик локальної мовної моделі для генерації відповіді*: Готова підказка (prompt) надсилається до розгорнутої локально LLM через програмний інтерфейс (API). Використовуємо локальний екземпляр моделі (наприклад, відкритої моделі Llama 2 від Meta) у вигляді Docker-сервісу, сумісного з API форматами OpenAI. Тобто запит до локальної моделі форматуємо за зразком запиту до OpenAI API (з зазначенням моделі, історії повідомлень тощо) і надсилаємо на локальний URL (наприклад, http://localhost:11434/api/generate). Модель опрацьовує підказку з контекстом і генерує підсумкову відповідь, яка містить інформацію з наданих фрагментів CSV (або повідомлення про те, що потрібні дані відсутні, якщо запит виходить за межі бази знань). Завдяки локальному розгортанню LLM, вся обробка відбувається без передачі даних

стороннім сервісам, що підвищує конфіденційність і автономність системи. Отримана від моделі відповідь повертається користувачеві чат-ботом.

4.1.2. Навчання і експериментальне дослідження.

Для реалізації LLM було використано згадану вище сучасну модель LLaMA 2. Використання розробленої моделі пояснюється потребою швидкої перевірки працездатності закладених у основу створення інтелектуального асистента підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП ідей.

Рішення, які приймалися на цьому етапі, визначалися основним завданням – швидко вибрати готове рішення з контролем кількох параметрів «розумного» помічника, який вміє відповідати на запитання на основі інформації зі звичайної таблиці (CSV-файлу) й використовувати сучасну велику мовну модель, яку можна розгорнути локально в Docker-контейнері.

Розглянемо деякі аспекти реалізації описаних вище кроків цього етапу дослідження, які реалізуються відповідними компонентами рішення:

1) *Підготовка та завантаження даних*: читаємо таблицю зі списком продуктів із CSV-файлу (як зазначалося вище була використана бібліотека Pandas для запису в структуру DataFrame);

2) *Обчислення семантичних векторних представлень для побудови бази знань*: а) перетворюємо кожний рядок таблиці на «короткий текст» (напр. «Модель: EXP2524. Дата завершення підтримки: 2020-06-30.»); б) зберігаємо ці описи у внутрішньому списку;

3) *Семантичний пошук релевантного контексту* (створення та пошук за семантичними вбудовуваннями): а) переводимо кожного тексту за допомогою моделі «all-MiniLM-L6-v2» в набір чисел (вектор); б) опрацьовуючи запит користувача також перетворюємо його у вектор і шукаємо найближчі за мірою подібності (косинусна подібність) описів;

4) *Формування підказки з контекстом для LLM* для передачі локальній мовній моделі: а) готуємо контекст (найбільш релевантні записи із таблиці);

5) *Виклик локальної мовної моделі для генерації відповіді*: а) відправляємо запит до локального сервера LLM через HTTP-запит у форматі,

сумісному з OpenAI API, або альтернативному форматі; б) отримуємо і повертаємо відповідь у вигляді тексту.

Розглянемо особливості виконання окремих кроків. Для навчання і перевірки ефективності LLM використано популярний набір даних про ЖЦ продуктів IBM. Використовувався спрощений робочий потік. Перший крок полягав у пошуку найближчих слів за прийнятими в LLaMA 2 метриками відстані. Сутністю другого кроку було підсумовування результатів пошуку. Генерування відповіді на основі підказки здійснювалося на кроці 4. Вміст підказки визначений концепціями проблемного підходу. Релевантний контекст підказки дозволив LLM частіше надавати правильні відповіді. Реалізація алгоритму неінформативного пошуку модифікацій не передбачала.

Експериментальне дослідження ґрунтувалося на аналізі відповідей моделі на питання вигляду: «Який продукт є найбільш актуальним на певну дату». Предметом аналізу спочатку були відповіді виключно компоненту LLM. Аналіз показав невисокий відсоток правильних відповідей – 40%. Інші відповіді містили неправдиву або неактуальну інформацію.

Потім умови експериментального дослідження були змінені – було виконане навчання системи і потім аналізувалися її відповіді на питання наведеного вище вигляду. Аналіз показав, що відсоток правильних відповідей зріс, але точність високого рівня забезпечити не вдалося.

Позитивного результату з огляду на точність вдалося досягти з використанням RAG-методології. Експеримент було продовжено шляхом аналізу відповідей компоненту LLM на питання наведеного вище вигляду, але з використанням підказок компоненту RAG. Результати показали, що у цьому випадку велика мовна модель значно рідше генерує неправдиву чи неактуальну інформацію. Загалом при застосуванні компоненту RAG неправдиві чи неактуальні відповіді були надані на близько 10% запитань. Отже, генерування відповідей LLM з використанням наданого компонентом RAG контексту суттєво поліпшує показник точності відповідей. Видається

справедливим припущення, що у цьому випадку відсоток правильних відповідей можна збільшити за рахунок додаткового навчання системи.

Експериментальне дослідження показало залежність ефективності взаємодії компонентів LLM і RAG від векторних представлень і метрик подібності. Раціональний баланс між швидкістю роботи та точністю результатів показала модель all-MiniLM-L6-v2.

Для невеликих та середніх за розміром CSV-файлів реалізація взаємодії компонентів LLM і RAG добре масштабується, але для великих CSV-файлів необхідна оптимізація. Щоб пришвидшити пошук доцільно використовувати спеціалізовані векторні БД або розбиття даних на тематичні блоки. Проте експеримент показав переваги підходу, адже вже простий прототип системи надає релевантну відповідь на запит користувача, спираючись на факти з CSV-файлу. Підтвердження ефективності взаємодії компонентів LLM і RAG загалом збагатилося можливістю створення інтелектуальних асистентів і аналітичних систем, здатних з бажаним рівнем точності і конфіденційності відповідати на питання з використанням внутрішніх даних організації.

Реалізований варіант взаємодії великої мовної моделі і RAG-системи з використанням даних у форматі CSV також демонструє ефективну інтеграцію структурованих даних з можливостями LLM. Метод семантичного пошуку з наступним генеративним поясненням відповідей є перспективним напрямом для побудови інтелектуальних систем запитання-відповідь, що поєднують структуровані бази знань і великі мовні моделі, здатні інтерпретувати та презентувати у зрозумілий спосіб користувачеві знання з цієї бази. Це підвищує ефективність використання накопичених даних і розширює сферу застосування LLM у наукових та прикладних задачах.

У реальній платформі підтримки процесів ЖЦ сервісів в ІС провайдерів ІКП для LLM і RAG будуть використовуватися проіндексовані БД і бази знань, в яких будуть використовуватися документи (текстові, графічні), продуктові і проектні матеріали, формальні засоби структурування накопичених даних, призначені для реалізації проблемного підходу на основі декомпозиції.

## 4.2 Розроблення нейронної мережі побудови архітектури сервісів

У розділі описується реалізація компонента побудови архітектури сервісів платформи підтримки ЖЦ сервісів з використанням нейромереж з глибоким навчанням, основи якого були закладені у праці [144].

### 4.2.1 Розроблення функціональної схеми генератора архтектури

Функціональна структура компонента платформи підтримки ЖЦ сервісів у ІС провайдерів ІКП, який відповідає за побудову архітектури сервісів, дозволяє повністю представити його функціональні можливості, оскільки містить як функції, так і їх взаємозв'язки, важливі з точки зору підтримки процесів побудови архітектури сервісів. Функціональна схема генератора архітектури наведена на рис. 4.1.

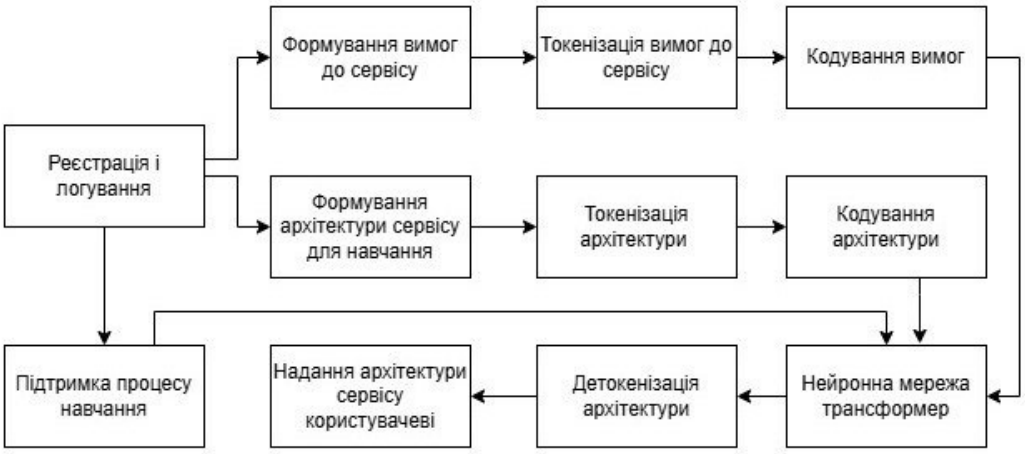


Рисунок 4.1 – Функціональна структура генератора архітектури сервісів

Тут кожна функція відображена відповідним елементом компонента. Виконана у відповідності принципу єдиної відповідальності функціональна схема дозволяє декомпозувати процеси ЖЦ сервісів на підпроцеси, а за необхідності навіть – на операції.

Для реалізації генератора архітектури, функціональна структура якого наведена вище, важливий матеріал надає діаграма випадків використання генератора, наведена на рис.4.2.

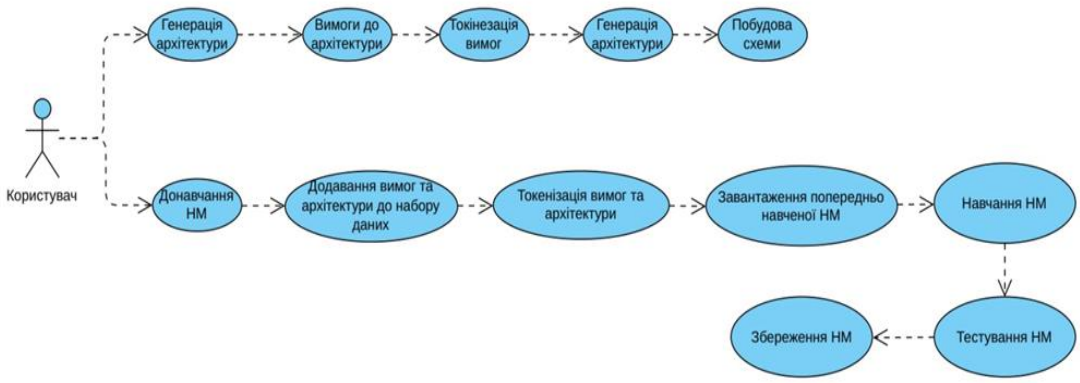


Рисунок 4.2 – Діаграма випадків використання генератора архітектури

### 4.2.2 Сценарії використання генератора архітектур сервісів

Розробивши схему роботи генератора архітектур сервісів, перейдемо до описання і аналізу сценаріїв його використання. Планується два сценарії роботи з даною програмою: робота по генерації самих архітектур по заданим параметрам; донавчання генератора.

У випадку застосування першого з них спочатку необхідно задати вимоги і потім отримати саму архітектуру. У випадку застосування другого сценарію спочатку необхідно задати пари – вимоги та архітектура. На наступному кроці задані пари додаються до набору даних, після чого виконується крок власне донавчання моделі з урахуванням нових даних.

### 4.2.3 Реалізація генератора архітектур сервісів

Щоб пришвидшити реалізацію генератора архітектур сервісів і виконати функціональні і нефункціональні вимоги були використані такі бібліотеки: import torch; import torch.nn as nn; import torch.optim as optim.

Розпочнемо з представлення вхідних даних. Вимоги до архітектури в закодованому вигляді подаються в X_data. Приклад заповненого X_data наведено у додатку Д. Зрозуміло, що результат також треба представляти у закодованому вигляді. Архітектура системи в закодованому вигляді подається в Y_data. Приклад заповненого X_data наведено у додатку Д.

Наступним кроком буде переведення даних в тензори. Перетворення вимог і архітектури в тензори наведено у додатку Д.

Наступним важливим кроком буде вбудовування позицій. Приклад коду вбудовування позицій наведено у додатку Д.

В рамках проєкту синтезу архітектури за допомогою нейронної мережі важливою складовою є модель. Приклад коду моделі наведено у додатку Д.

Тепер перейдемо до тренування нейронної мережі. Тут необхідно врахувати всі описані вище рішення щодо її структури та інструментів:

```
model = SimpleTransformer()
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
start_token = 3
tgt_input = torch.full((1,1), start_token, dtype=torch.long)  # (batch, 1)
#Функція, щоб послідовно подавати цілі у декодер (teacher forcing)
def train_step(x, y):
    model.train()
    optimizer.zero_grad()
    #Для трансформера decoder input – зсунуті цілі на 1 позицію вправо
    tgt_inp = torch.cat([tgt_input, y[:, :-1]], dim=1)  # (batch, tgt_len)
    output = model(x, tgt_inp)  # (batch, tgt_len, vocab_size)
    loss = criterion(output.view(-1, vocab_size), y.view(-1))
    loss.backward()
    optimizer.step()
    return loss.item()
```

Насамкінець, опишемо навчання нейронної мережі:

```
epochs = 2700
for epoch in range(epochs):
    loss = train_step(x, y)
    if (epoch+1) % 50 == 0:
        print(f"Epoch {epoch+1}, Loss: {loss:.10f}")
```

Дані цього кроку дослідження можливостей нейромережі з глибоким навчанням для синтезу архітектури сервісів наведено у додатку Е.

Тепер знову тестуємо нейронну мережу:

```
model.eval()
with torch.no_grad():
    tgt_inp = torch.full((1,1), start_token, dtype=torch.long)
    outputs = []
    for _ in range(output_seq_len):
        out = model(x, tgt_inp)  # (1, len(tgt_inp), vocab_size)
        next_token = out[:, -1, :].argmax(dim=-1, keepdim=True)  # (1,1)
        tgt_inp = torch.cat([tgt_inp, next_token], dim=1)
        outputs.append(next_token.item())
    #Перетворюємо назад у матрицю 21x21
    predicted = torch.tensor(outputs).reshape(21,21)
    print("\nПередбачено (Y_data):")
    print(predicted)
    print("\nЗначення за умовою (Y_data):")
    print(torch.tensor(Y_data))
```

Результати підтверджують правильність закладених рішень і придатність генерованих нейронною мережею архітектур сервісів.

### 4.3 Реалізація компонента групування сервісів у пакети

Будемо дотримуватись ЖЦ створення ПЗ ІС. Аналіз процесів підтримки процесів ЖЦ сервісів, насамперед групування сервісів у пакети, виконаний у попередніх підрозділах. Функціональні вимоги до компоненту групування сервісів у пакети наведені у підрозділі 3.5. Там же наведені нефункціональні вимоги до системи, зокрема здатність до розвитку та інші. Тому перейдемо до побудови архітектури компоненту. Функціональні вимоги до компоненту дозволять визначити відповідні функціональні елементи архітектури. Такий елемент представлятиме кожний запропонований метод і алгоритм розв’язання задачі групування сервісів у пакети і метод урахування опіній бізнес- і ІТ-менеджерів. Жорсткі нефункціональні вимоги до системи в цілому

і компоненту, його належність до складної платформи підтримки ЖЦ сервісів у ІС провайдерів ІКП визначають вибір мікросервісної архітектури. Тоді кожному згаданому вище методу і алгоритму відповідатиме функціональний мікросервіс. Традиційні для мікросервісної архітектури точка входу і мікросервіс аутентифікації забезпечать взаємодію функціональних елементів, щоб надати компоненту бажаної поведінки. Ефективне використання хмарних ресурсів (у нашому випадку Azure) забезпечить OKTA/Azure AD. Усі мікросервіси будуть працювати є однією БД за допомогою MS SQL Server. Додавши клієнтську частину отримуємо наведену на рис. 4.3 архітектуру.

Побудована архітектура дає бачення створюваного компоненту платформи, яке буде покладене в основу реалізації компоненту. Для реалізації компоненту обґрунтовано вибрані мови, фреймворки, технології і бібліотеки. Клієнтська частини в рамках взаємодії бізнес- і IT-менеджерів з компонентом підтримує організацію їх діяльності зі збору, введення, колективного опрацювання опіній і використання результатів. Для цього вона відображає можливості користувача і взаємодіє з серверною частиною. Реалізована взаємодія з користувачем дозволяє ініціювати процеси формування портфелів сервісів, які IT-компанії надають провайдерам і провайдери – своїм клієнтам.

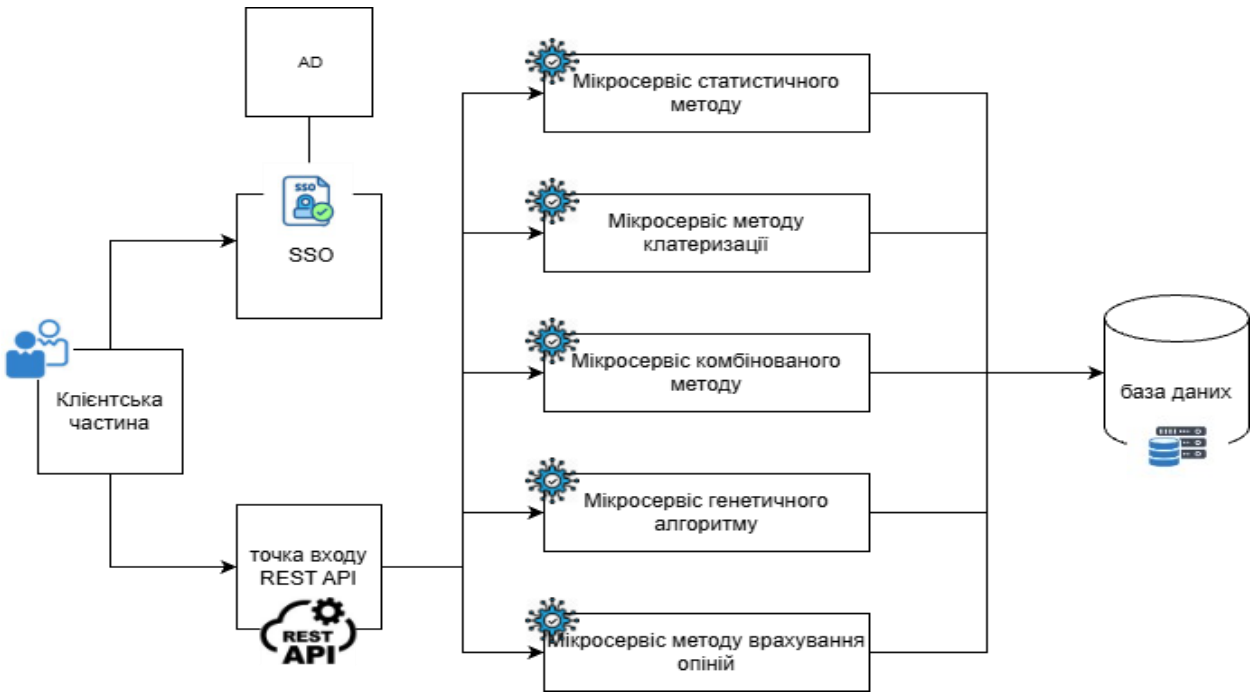


Рисунок 4.3 – Архітектура компонента групування сервісів у пакети

Клієнтська частина надає бізнес- і ІТ-менеджерам зручний інтерфейс для використання мікросервісів групування сервісів, перегляду пакетів сервісів, які ІТ-компанії надавали провайдерам, а провайдери – своїм клієнтам, формування опіній, оцінювання і приймання рішень щодо пакетів, вибору найбільш придатних методів за різними параметрами, редагувати рішення, дані ОПР, аналізувати відповідність їх опіній ситуаціям, які їх вимагали. Клієнтська частина реалізована на фреймворку React, взаємодія з системою – через HTTP-запити. Мікросервіс одноразової автентифікації, базований на зовнішньому сервісі OKTA/MS AzureAD, забезпечує захист ресурсів системи і уможливлює ініціювання потрібних процесів за допомогою токена доступу.

Доступ до основних функцій системи здійснюється через точку входу API. Вона отримує запити користувачів, пересилає їх мікросервісам з врахуванням структури системи і описів мікросервісів, поєднаних у реєстрі. Бізнес- чи ІТ-менеджер, залежно від ситуації та обов'язків, має можливість:

1) викликати мікросервіс методу (статистичного чи комбінованого), алгоритму (кластеризації, удосконаленого керованого генетичного алгоритму) групування сервісів у пакет для вибраного провайдера чи клієнта провайдера;

2) викликати мікросервіс методу урахування опіній бізнес- і ІТ-менеджерів, щоб проаналізувати результати застосування мікросервісів для відповідного методу групування сервісів у пакети, напрацювати рекомендації щодо найкращого методу (остаточне рішення за ОПР).

З використанням реалізованого компоненту групування сервісів у пакети було виконане експериментальне дослідження працездатності методів групування сервісів у пакети, а також методу урахування опіній бізнес- і ІТ-менеджерів для розв'язання задачі групування сервісів у пакети. Дані для дослідження методу урахування опіній бізнес- і ІТ-менеджерів представлені у табл. 4.1 (це результат заповнення опініями бізнес- і ІТ-менеджерів таблиці, представленої на рисунку 3.7). Мікросервіс урахування опіній бізнес- і ІТ-менеджерів, реалізований на основі методу наведеного у праці [72] дав такі результати: вектор $H$ = (0.37, 0.34, 0.28) (його компоненти – нормалізовані

важливості критеріїв); вектор $E(P,C) = (0.015, 0.021, 0.013, 0.03)$ (його компонентами виступають відстані між розподілами ймовірностей).

Отримані дані зведення опіній, з одного боку, підтвердили у цій ситуації перевагу методу М3, якому відповідає найменша відстань 0.013, з іншого боку, підтвердили працездатність запропонованого методу урахування опіній бізнес- і ІТ-менеджерів. Отже, результати експериментального дослідження підтвердили доцільність використання методу у відповідному компоненті платформи підтримки ЖЦ сервісів в ІС провайдерів інфокомунікацій.

Таблиця 4.1 – Приклад опіній бізнес- і ІТ-менеджерів для експерименту

| Бізнес- чи ІТ-менеджер | Упорядкування критеріїв | Опінія щодо методу $A_1$ за критеріями $K_1$, $K_2$, $K_3$ | | | Опінія щодо методу $A_2$ за критеріями $K_1$, $K_2$, $K_3$ | | | Опінія щодо методу $A_3$ за критеріями $K_1$, $K_2$, $K_3$ | | | Опінія щодо методу $A_4$ за критеріями $K_1$, $K_2$, $K_3$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 1 | 2, 3, 1 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 |
| 2 | 3, 2, 1 | 9 | 7 | 0 | 6 | 5 | 7 | 5 | 4 | 2 | 4 | 5 | 7 |
| 3 | 2, 1, 3 | 6 | 6 | 8 | 4 | 3 | 2 | 7 | 7 | 6 | 8 | 9 | 0 |
| 4 | 1, 2, 3 | 5 | 4 | 6 | 1 | 2 | 3 | 7 | 6 | 5 | 8 | 7 | 9 |
| 5 | 1, 3, 2 | 8 | 7 | 9 | 5 | 2 | 4 | 8 | 8 | 9 | 3 | 3 | 5 |
| 6 | 3, 1, 2 | 8 | 9 | 0 | 4 | 3 | 6 | 2 | 3 | 2 | 6 | 5 | 8 |
| 7 | 1, 2, 3 | 8 | 6 | 8 | 5 | 5 | 6 | 7 | 8 | 9 | 6 | 6 | 8 |
| 8 | 2, 1, 3 | 5 | 6 | 8 | 1 | 3 | 2 | 7 | 8 | 0 | 6 | 7 | 8 |
| 9 | 1, 2, 3 | 7 | 6 | 5 | 8 | 7 | 8 | 6 | 4 | 4 | 4 | 5 | 6 |

## 4.4 Особливості використання платформи підтримки ЖЦ сервісів

### 4.4.1 Особливості природномовної комунікації

Реалізація функцій природномовної комунікації платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП виконується в рамках компоненту LLM з використанням моделі LLaMA 2, що дозволило розробити з інструмент з

широкими можливостями. Модель LLaMA 2 надає інструменти для реалізації генерування відповідей на основі підказок розробленого компоненту RAG, що значно спрощує реалізацію запропонованої платформи.

4.4.2 Особливості побудови архітектури сервісів

Реалізація функцій побудови архітектури сервісів запропонованої платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП виконується в рамках розробленої нейронної мережі з глибоким навчанням. Вона надає інструменти для введення і кодування вхідних даних провайдера, навчання мережі на розроблених архітектурах для провайдерів з визначеними вхідними даними, генерування архітектури на основі побудованого мережею коду. Це дозволяє поліпшувати архітектури на основі накопичених результатів роботи нейромережі.

4.4.3 Особливості застосування компоненту групування сервісів у пакети

Реалізація функцій групування сервісів у пакети запропонованої платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП виконується в рамках розробленого компоненту групування сервісів у пакети з відповідними інструментами. Компонент надає інструменти у вигляді мікросервісів для групування сервісів у пакети на основі чотирьох методів, які враховують особливості конкретної ситуації, та мікросервіс для врахування опіній бізнес- і ІТ-менеджерів і вибору найбільш придатного. Перевагою реалізованого компоненту групування сервісів у пакети є його універсальність, що дозволяє використовувати функціонал пакету для різних провайдерів.

**Висновки до розділу 4**

Реалізовано прототип платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП на основі запропонованих в дисертаційній роботі моделей та методів. Виконано експериментальну перевірку здатності прототипу платформи підтримки ЖЦ сервісів до скорочення термінів розроблення і впровадження ІС провайдерів ІКП, зниження витрат на їх розроблення,

впровадження та операційну підтримку, а також покращення їх продуктивності, відмовостійкості, придатності, розширюваності та інших показників якості за рахунок реалізації на основі запропонованих в роботі моделей і методів інструментів автоматизації процесів підтримки ЖЦ сервісів.

В рамках розробленого прототипу платформи реалізовано і експериментально перевірено компоненти великої мовної моделі і RAG-системи для забезпечення комунікації користувачів в контексті проблем, які вони розв’язують в ІС провайдерів ІКП.

Також розроблено і експериментально перевірено низку компонентів етапів ініціації, виявлення вимог і проєктування сервісів. Зокрема для вибору найкращих розв’язків задачі групування сервісів у пакети реалізовані статистичний метод, метод кластеризації, удосконалені варіант керованого генетичного алгоритму і комбінованого методу розв’язання змішаних задач математичного програмування, підхід до врахування опіній бізнес- і ІТ-менеджерів. Експеримент підтвердили їх працездатність, здатність групувати сервіси у економічно вигідні для всіх сторін пакети, можливість узгоджувати точність і обчислювальні витрати, використання в умовах невизначеності і обмежених ресурсів за рахунок технологій управління ресурсами.

Загалом реалізація і експериментальне дослідження прототипу платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП підтвердило їх здатність до автоматизації процесів ЖЦ сервісів при створенні, експлуатації і розвитку ІС провайдерів ІКП. Зокрема експериментально підтверджено, що використання прототипу платформи дозволяє скоротити терміни розроблення і впровадження ІС провайдерів ІКП, одночасно уможливлюючи зниження витрат на їх розроблення, впровадження та операційну підтримку. Крім того, автоматизація робіт з розроблення, впровадження та операційної підтримки ІС провайдерів ІКП також приводить до покращення їх показників якості. Машинне навчання, нейромережі, мультиагентні системи, інтелектуальний аналіз, інтелектуальний моніторинг, інші концепції ШІ, моделювання з використанням сучасних формальних моделей теорії прийняття рішень, математичного програмування підтвердили свою практичну спрямованість.

# ВИСНОВКИ

У дисертаційній роботі вирішується актуальна проблема підтримки ЖЦ сервісів в ІС провайдерів ІКП. Виконаний аналіз сучасного стану, моделей і методів підтримки ЖЦ сервісів та підходів до проєктування сервісів і групування сервісів у пакети в ІС провайдерів ІКП показав, що постає потреба в автоматизації процесів ЖЦ сервісів в ІС провайдерів ІКП.

На основі проведеного аналізу поточного стану і тенденцій впровадження сервісного підходу в ІС провайдерів ІКП встановлено, що саме платформа підтримки ЖЦ сервісів шляхом автоматизації зазначених процесів забезпечить, з одного боку, скорочення термінів створення ІС, зменшення витрат на їх створення, впровадження і експлуатацію, з іншого боку, підвищення показників їх продуктивності, відмовостійкості, придатності та розширюваності, що гарантуватиме надання необхідного рівня якості послуг провайдерами в сучасних умовах розвитку галузі.

Обґрунтовано вимоги до платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП та розроблено принципи її побудови, реалізації і застосування, розроблено архітектуру платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП, яка базується на моделі End-to-End та використовує поєднання моделей штучного інтелекту, математичного програмування і теорії прийняття рішень для підтримки процесів проєктування, реалізації, експлуатації та надання сервісів, що дозволяє скоротити терміни впровадження ІС, зменшити витрати на їх розгортання та операційну підтримку, а також підвищити показники продуктивності, відмовостійкості, придатності та розширюваності.

Доведено, що підтримка ЖЦ сервісів вимагає координації працівників ІТ-компаній і провайдерів в контексті розуміння проблем та інтеграції сучасних інструментів реалізації процесів ЖЦ сервісів. Обґрунтовано необхідність використання сучасних технологій штучного інтелекту для природномовної комунікації з ІС працівників ІТ-компаній і провайдерів в

контексті розуміння проблем, які перед ними постають, а також розроблення та впровадження моделей і методів реалізації процесів ЖЦ сервісів в рамках вирішення зазначених проблем.

Запропоновано архітектуру інтелектуального асистента для організації і підтримки процесів ЖЦ сервісів як інтегральної основи розуміння і розв'язання проблем бізнес-діяльності ІТ-компанії і провайдерів і генерації відповідей з використанням комплексу великих мовних моделей і RAG-систем, які забезпечують взаємодію компонентів платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП в умовах природномовної взаємодії з виконавцями в рамках проблемного контексту з використанням накопичених текстових, табличних і графічних даних за схемою взаємодії інтелектуальних агентів для підключення інструментів платформи для розв'язання проблем користувачів.

Запропоновано метод формування архітектури сервісів в ІС провайдерів ІКП, який відрізняється структурою і набором елементів нейромережевої моделі з глибоким навчанням типу трансформер і підходом до її навчання, що дозволило використовувати її як інструмент синтезу архітектурних рішень для сервісів, які забезпечують ефективне використання ресурсів з урахуванням параметрів сервісів, вимог провайдерів та характеристик множини їх клієнтів.

Удосконалено комбінований метод групування сервісів у пакети, який відрізняється поєднанням декомпозиції задачі на підзадачі ІТ-компанії і провайдерів ІКП, ймовірнісно-жадібного алгоритму і алгоритму мурашиної колонії, евристичного пошуку компромісу, що дозволяє групувати сервіси у взаємовигідні пакети сервісів в різних умовах з урахуванням системи знижок до преференційної ціни, специфічних особливостей провайдера і його клієнтів, ресурсів ІТ-компанії, сучасних концепцій ведення бізнесу, накопиченого досвіду, гнучко регулювати баланс між дослідженням нових комбінацій і посиленням вже успішних розв'язків, підтримуючи самоорганізоване й адаптивне навчання.

Дістав подальшого розвитку керований генетичний алгоритм, який відрізняється додатковими параметрами двостороннього тиску відбору і

відносної швидкості наближення до оптимуму та правилом знижок, введеними для поєднання відомої ідеї управління збіжністю алгоритму і нової концепції відбору розв'язків з врахуванням багатокритеріальності, що забезпечує досягнення глобального оптимуму при розв'язанні задачі групування сервісів у пакети з цільовими функціями для ІТ-компанії і провайдерів за рахунок зростання значень обох функцій або компенсації зменшення значення однієї функції більшим зростанням іншої, що дозволило отримувати близькі до оптимальних рішення за умови виконання обмежень. Наявність удосконалених комбінованого методу і керованого генетичного алгоритму розв'язання задачі групування сервісів у пакети надала можливість вибирати метод для моделі змішаного програмування з врахуванням часу, відведеного на прийняття рішення.

Розроблений і реалізований прототип платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП, яка відрізняється базуванням на моделі E2E, проведено дослідження його функціональних можливостей та оцінювання ефективності використання отриманих результатів.

Розроблені в роботі моделі і методи реалізовані у вигляді програмних компонентів прототипу платформи підтримки ЖЦ сервісів в ІС провайдерів ІКП. Виконане експериментальне дослідження підтвердило працездатність платформи і її компонентів, довело ефективність запропонованих рішень.

Дані експериментів підтвердили здатність платформи сприяти автоматизації процесів ЖЦ сервісів в ІС провайдерів ІКП, що дозволило скоротити терміни впровадження ІС, зменшити витрати на їх впровадження та операційну підтримку і поліпшити їх продуктивність, відмовостійкість, придатність, розширюваність та інші якісні показники.

Виконане експериментальне дослідження дозволяє передбачити високу ефективність сучасних моделей і методів генерації програм, функціонального і логічного програмування, інтелектуального аналізу і моніторингу, моделювання, прогнозування для автоматизації процесів інших етапів ЖЦ сервісів, зокрема реалізації, тестування, впровадження, надання, розвитку.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Field J. M., Fotheringham D., Subramony M., Gustafsson A., Ostrom A. L., Lemon K. N., McColl-Kennedy J. R. Service research priorities: designing sustainable service ecosystems // *Journal of Service Research*. 2021. Vol. 24, No. 4. P. 462–479. DOI: 10.1177/10946705211031302.

2. Favoretto C., Mendes G. H. S., Oliveira M. G., Cauchick-Miguel P. A., Coreynen W. From servitization to digital servitization: How digitalization transforms companies' transition towards services // *Industrial Marketing Management*. 2022. Vol. 102. P. 104–121. DOI: 10.1016/j.indmarman.2022.01.003.

3. Kowalkowski C., Gebauer H., Kamp B., Parry G. Servitization and deservitization: Overview, concepts, and definitions // *Industrial Marketing Management*. 2017. Vol. 60. P. 4–10. DOI: 10.1016/j.indmarman.2016.12.007.

4. Rabetino R., Harmsen W., Kohtamäki M., Sihvonen J. Structuring servitization-related research // *International Journal of Operations & Production Management*. 2018. Vol. 38, No. 2. P. 350–371. DOI: 10.1108/IJOPM-03-2017-0175.

5. Zheng P., Wang Z., Chen C. H., Khoo L. P. A survey of smart product-service systems: Key aspects, challenges and future perspectives // *Advanced Engineering Informatics*. 2019. Vol. 42. Art. 100973. DOI: 10.1016/j.aei.2019.100973.

6. Zaki M. Digital transformation: harnessing digital technologies for the next generation of services // *Journal of Services Marketing*. 2019. Vol. 33, No. 4. P. 429–435. DOI: 10.1108/JSM-01-2019-0034.

7. Gomez-Trujillo A. M., Gonzalez-Perez M. A. Digital transformation as a strategy to reach sustainability // *Smart and Sustainable Built Environment*. 2022. Vol. 11, No. 4. P. 1137–1162. DOI: 10.1108/SASBE-01-2021-0011.

8. Farayola O. A., Hassan A. O., Adaramodu O. R., Fakeyede O. G., Oladeinde M. Configuration management in the modern era: best practices,

innovations, and challenges // *Computer Science & IT Research Journal*. 2023. Vol. 4, No. 2. P. 140–157. DOI: 10.51594/csitrj.v4i2.613.

9. Serrano J., Faustino J., Adriano D., Pereira R., da Silva M. M. An IT service management literature review: challenges, benefits, opportunities and implementation practices // *Information*. 2021. Vol. 12, No. 3. Art. 111. DOI: 10.3390/info12030111.

10. Richter H. D., Lantow B. IT-service value modeling: a systematic literature analysis // *Business Information Systems Workshops: BIS 2021*. Lecture Notes in Business Information Processing. Cham : Springer, 2022. Vol. 444. P. 267–278. DOI: 10.1007/978-3-031-04216-4_24.

11. Mora M., Marx-Gomez J., Wang F., Diaz O. Agile IT Service Management Frameworks and Standards: A Review // *Advances in Software Engineering, Education, and e-Learning*. Cham : Springer, 2021. P. 921–936. DOI: 10.1007/978-3-030-70873-3_66.

12. Tuunanen T., Lumivalo J., Vartiainen T., Zhang Y., Myers M. D. Micro-Level Mechanisms to Support Value Co-Creation for Design of Digital Services // *Journal of Service Research*. 2023. DOI: 10.1177/10946705231173116.

13. Schmidt M., Brenner M., Schaaf T. IT Service Management Frameworks Compared – Simplifying Service Portfolio Management // *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. 2019. P. 421–427. [Електронний ресурс]. Режим доступу: (IFIP Open Digital Library PDF). Дата звернення: 10.10.2024.

14. Agutter C. *ITIL Foundation Essentials ITIL 4 Edition: The Ultimate Revision Guide*. IT Governance Publishing, 2020.

15. Wirtz J., Fritze M. P., Jaakkola E., Gelbrich K., Hartley N. Service products and productization // *Journal of Business Research*. 2021. Vol. 137. P. 411–421. DOI: 10.1016/j.jbusres.2021.08.033.

16. Tuunanen T., Salo M., Li F. Modular service design of information technology-enabled services // *Journal of Service Research*. 2023. Vol. 26, No. 2. P. 270–282. DOI: 10.1177/10946705221082775.

17. OGC. *The Official Introduction to the ITIL Service Lifecycle*. London : The Stationery Office, 2007.

18. Van Bon J., de Jong A., Kolthof A., Pieper M., Tjassing R., van der Veen A., Verheijen T. *Foundations of IT Service Management Based on ITIL®*. Vol. 3. Van Haren Publishing, 2008. ISBN 978-9087532284.

19. Управління корпоративною IT-інфраструктурою / О.І. Ролік, С.Ф. Теленик, М.В. Ясочка // К.: Наукова думка, 2018. – 576 с. ISBN: 978-966-00-1647-7

20. Zharikov E., Telenyk S., Bidyuk P. Adaptive workload forecasting in cloud data centers // *Journal of Grid Computing*. 2020. Vol. 18. P. 149–168. DOI: 10.1007/s10723-019-09501-2.

21. What is ITSM? [Електронний ресурс]. – Режим доступу: https://www.servicenow.com/products/itsm/what-is-itsm.html (дата звернення: 01.04.2023).

22. Koivisto J., Hamari J. The rise of motivational information systems: A review of gamification research // *International Journal of Information Management*. 2019. Vol. 45. P. 191–210. DOI: 10.1016/j.ijinfomgt.2018.10.013.

23. Stair R., Reynolds G. *Principles of Information Systems*. 14th ed. Boston: Cengage Learning, 2020. ISBN 978-0357112410.

24. Gregor S., Chandra Kruse L., Seidel S. Research perspectives: The anatomy of a design principle // *Journal of the Association for Information Systems*. 2020. Vol. 21, No. 6. P. 1622–1652. DOI: 10.17705/1jais.00649.

25. Service Management. [Електронний ресурс]. – Режим доступу: https://www.sap.com/products/service-management.html (дата звернення: 01.04.2023).

26. Automation & Zero-touch IT: The Keys to Optimizing SaaS [Електронний ресурс]. – Режим доступу: https://www.spiceworks.com/tech/innovation/guest-article/zero-touch-it-optimizing-saas/ (дата звернення: 01.04.2023).

27. Kohli R., Melville N. P. Digital innovation: A review and synthesis // *Information Systems Journal*. 2019. Vol. 29, No. 1. P. 200–223. DOI: 10.1111/isj.12193.

28. Sallé M. IT Service Management and IT Governance: review, comparative analysis, and their impact on utility computing. HP Laboratories Palo Alto, 2004. P. 8–17. [Електронний ресурс]. – Режим доступу: http://www.hpl.hp.com/techreports/2004/HPL-2004-98.pdf (дата звернення: 01.04.2023).

29. Capability Maturity Model Integration (CMMI) [Електронний ресурс]. – Режим доступу: https://cmmiinstitute.com/products (дата звернення: 01.04.2023).

30. CMMI Institute – Resources [Електронний ресурс]. – Режим доступу: https://cmmiinstitute.com/resources (дата звернення: 01.04.2023).

31. TM Forum. DTW Ignite [Електронний ресурс]. – Режим доступу: https://dtw.tmforum.org/ (дата звернення: 17.05.2025).

32. Maddern H., Smart P. A., Maull R. S., Childe S. End-to-end process management: implications for theory and practice // *Production Planning & Control*. 2014. Vol. 25, No. 16. P. 1303–1321. DOI: 10.1080/09537287.2013.832821.

33. Obbink H., America P. Towards evergreen architectures: on the usage of scenarios in system architecting // *Proceedings of the International Conference on Software Maintenance (ICSM 2003)*. IEEE, 2003. P. 298–307. DOI: 10.1109/ICSM.2003.1235435.

34. Mohr W., Konhäuser W. Access network evolution beyond third generation mobile communications // *IEEE Communications Magazine*. 2000. Vol. 38, No. 12. P. 122–133. DOI: 10.1109/35.888266.

35. Eikerling H. J., Mazzoleni P., Plaza P., Yankelevich D., Wallet T. Services and mobility: the PLASTIC answer to the Beyond 3G challenge : white paper. PLASTIC Project, 2007.

36. Service platform for innovative communication environment (SPICE), Grant agreement ID: 027617 [Електронний ресурс]. – Режим доступу: https://cordis.europa.eu/project/id/027617 (дата звернення: 01.04.2023).

37. Open platform for user-centric service creation and execution (OPUCE), Grant agreement ID: 034101 [Електронний ресурс]. – Режим доступу: https://cordis.europa.eu/project/id/034101 (дата звернення: 01.04.2023).

38. Top 7 Integration Platform as a Service (iPaaS) Software Companies in 2021 [Електронний ресурс]. – Режим доступу: https://www.spiceworks.com/tech /cloud/articles/top-ipaas-companies/ (дата звернення: 01.04.2023).

39. What Are Managed IT Services and Why Do You Need Them? [Електронний ресурс]. – Режим доступу: https://www.spiceworks.com/tech /cloud/articles/managed-it-services/ (дата звернення: 01.04.2023).

40. Pardo C., Pagani M., Savinien J. The strategic role of social media in business-to-business contexts // *Industrial Marketing Management*. 2022. Vol. 101. P. 82–97. DOI: 10.1016/j.indmarman.2021.11.010.

41. Roy R. R. *Handbook of SDP for Multimedia Session Negotiations: SIP and WebRTC IP Telephony*. Boca Raton : CRC Press, 2018. DOI: 10.1201/9781351023887.

42. Lopes S. F. The importance of the ITIL framework in managing information and communication technology services // *International Journal of Advanced Engineering Research and Science*. 2021. Vol. 8, No. 5. P. 292–296. DOI: 10.22161/ijaers.85.37.

43. Hietanen S. Mobility as a Service – the new transport model // *Eurotransport*. 2014. Vol. 12, No. 2. P. 2–4.

44. Esztergár-Kiss D., Kerényi T. Creation of mobility packages based on the MaaS concept // *Travel Behaviour and Society*. 2020. Vol. 21. P. 307–317. DOI: 10.1016/j.tbs.2020.07.007.

45. Dutt A., Jain H., Kumar S. Providing software as a service: a design decisions model // *Information Systems and e-Business Management*. 2018. Vol. 16. P. 327–356. DOI: 10.1007/s10257-017-0354-2.

46. TM Forum Service Delivery Framework (SDF) [Електронний ресурс]. – Режим доступу: https://studylib.net/doc/8567280/tm-forum-service-delivery-framework--sdf- (дата звернення: 01.10.2023).

47. Global Service Delivery Platform Market Growth, Trends, and Forecast 2019–2024 – Platform-as-a-Service (PaaS) is Expected to Hold the Largest Share // *BusinessWire*. 2019. [Електронний ресурс]. – Режим доступу: https://www.businesswire.com/ (дата звернення: 01.10.2023).

48. Hyötyläinen M., Möller K. Service packaging: key to successful provisioning of ICT business solutions // *Journal of Services Marketing*. 2007. Vol. 21, No. 5. P. 304–312. DOI: 10.1108/08876040710773690.

49. Beynon-Davies P. *Business Information Systems*. London : Bloomsbury Publishing, 2019. ISBN 9781350304741.

50. Tilley S. *Systems Analysis and Design*. Boston : Cengage Learning, 2019. ISBN 9781337100010.

51. Peppard J. Managing IT as a portfolio of services // *European Management Journal*. 2003. Vol. 21, No. 4. P. 467–483. DOI: 10.1016/S0263-2373(03)00074-4.

52. Shostack G. L. How to design a service // *European Journal of Marketing*. 1982. Vol. 16, No. 1. P. 49–63. DOI: 10.1108/EUM0000000004799.

53. Teixeira J. G., Patrício L., Tuunanen T. Bringing design science research to service design // *Exploring Service Science : IESS 2018 Proceedings*. Cham : Springer, 2018. P. 373–384. DOI: 10.1007/978-3-319-98512-1_32.

54. Glushko R. J., Nomorosa K. J. Substituting information for interaction: a framework for personalization in service encounters and service systems // *Journal of Service Research*. 2013. Vol. 16, No. 1. P. 21–38. DOI: 10.1177/1094670512467917.

55. Ahmad A. A., Arshah R. A., Kamaludin A., Ngah L., Bakar T. A., Zakaria M. R. Adopting service level agreement (SLA) in enhancing the quality of IT hardware service support // *International Journal of Synergy in Engineering and Technology*. 2020. Vol. 1, No. 1.

56. Jackson D. Alloy: a language and tool for exploring software designs // *Communications of the ACM*. 2019. Vol. 62, No. 9. P. 66–76. DOI: 10.1145/3338846.

57. Tsui F., Karam O., Bernal B. *Essentials of Software Engineering*. Burlington : Jones & Bartlett Learning, 2022. ISBN 9781284230543.

58. Telenyk S., Nowakowski G., Zharikov E., Vovk Y. Information technology for web-applications design and implementation // *Адаптивні системи автоматичного управління*. 2019. No. 1(34). P. 138–151.

59. MacLean D., Titah R. Implementation and impacts of IT service management in the IT function // *International Journal of Information Management*. 2023. Vol. 70. Art. 102628. DOI: 10.1016/j.ijinfomgt.2023.102628.

60. Yandri R., Utama D. N., Zahra A. Evaluation model for the implementation of information technology service management using fuzzy ITIL // *Procedia Computer Science*. 2019. Vol. 157. P. 290–297. DOI: 10.1016/j.procs.2019.08.175.

61. Hans A. R., Firmansyah G., Tjahyono B., Widodo A. M. Evaluation of IT service level infrastructure in organizations using ITIL version 3 standardization // *Asian Journal of Social and Humanities*. 2024. Vol. 2, No. 12. P. 2995–3006.

62. Başar A. IT service level optimization: a case study // *Proceedings of the International Conference on Industrial Engineering and Operations Management*. Istanbul, 2022. P. 421–427. [Електронний ресурс]. – Режим доступу: https://www.researchgate.net/publication/362061884 (дата звернення: 15.02.2026).

63. Brocke H., Uebernickel F., Brenner W. A methodical procedure for designing consumer-oriented on-demand IT service propositions // *Information Systems and e-Business Management*. 2011. Vol. 9, No. 2. P. 283–302. DOI: 10.1007/s10257-010-0148-5.

64. Zhu Z. Y., Xie H. M., Chen L. ICT industry innovation: knowledge structure and research agenda // *Technological Forecasting and Social Change*. 2023. Vol. 189. Art. 122361. DOI: 10.1016/j.techfore.2023.122361.

65. Li X., Chen C. H., Zheng P., Wang Z., Jiang Z., Jiang Z. A knowledge graph-aided concept-knowledge approach for evolutionary smart product-service

system development // *Journal of Mechanical Design*. 2020. Vol. 142, No. 10. Art. 101403. DOI: 10.1115/1.4046922.

66. The TOGAF® Standard, Version 9.2 [Електронний ресурс]. – Режим доступу: https://www.opengroup.org/togaf-standard-version-92-overview (дата звернення: 01.04.2023).

67. The ArchiMate® Enterprise Architecture Modeling Language [Електронний ресурс]. – Режим доступу: https://www.opengroup.org/archimate-forum/archimate-overview (дата звернення: 01.04.2023).

68. Чимшир В., Теленик С., Ролік О., Жаріков Е. Платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг // *Адаптивні системи автоматичного управління*. 2023. Vol. 1, No. 42. DOI: 10.20535/1560-8956.42.2023.279172.

69. Гавриленко О., Чимшир В., Жаріков Е., Теленик С., Омельченко Р. Вирішення задачі впровадження пакетів сервісів за допомогою статистичної інформації // *Адаптивні системи автоматичного управління*. 2023. Vol. 2, No. 43. P. 94–107. DOI: 10.20535/1560-8956.43.2023.292257.

70. Гавриленко О., Чимшир В., Жаріков Е., Теленик С., Амонс О. Метод формування пакетів сервісів за допомогою алгоритмів кластеризації // *Адаптивні системи автоматичного управління*. 2024. Vol. 1, No. 44. P. 182–191. DOI: 10.20535/1560-8956.44.2024.302437.

71. Chymshyr V., Zhdanova O., Havrylenko O., Nowakowski G., Telenyk S. Models and methods for forming service packages for solving the problem of designing services in information systems of providers // *Information, Computing and Intelligent Systems Journal*. 2024. No. 5. P. 29–54. DOI: 10.20535/2786-8729.5.2024.316432.

72. Чимшир В., Омельченко Р. Вибір методу формування портфеля сервісів для провайдерів інфокомунікацій з урахуванням ситуації // *Наукові праці ВНТУ*. 2025. No. 3. P. 1–12. DOI: 10.31649/2307-5376-2025-3-149-160.

73. Жданова О., Букасов М., Цимбал С., Чимшир В., Омельченко Р. Інформаційна система формування пакетів сервісів для провайдерів

інфокомунікацій // *Computer Systems and Information Technologies*. 2025. No. 3. P. 99–114. DOI: 10.31891/csit-2025-3-10.

74. Spohrer J., Kwan S. Service science, management, engineering, and design (SSMED): an emerging discipline – outline & references // *International Journal of Information Systems in the Service Sector*. 2009. Vol. 1, No. 3. DOI: 10.4018/jisss.2009070101.

75. Oláh J., Kitukutha N., Haddad H., Pakurár M., Máté D., Popp J. Achieving sustainable e-commerce in environmental, social and economic dimensions by taking possible trade-offs // *Sustainability*. 2019. Vol. 11. Art. 89. DOI: 10.3390/su11010089.

76. Morelli N., de Götzen A., Simeone L. *Service Design Capabilities*. Springer Series in Design and Innovation. Cham : Springer, 2020. DOI: 10.1007/978-3-030-56282-3.

77. Daidj N. Towards renewed business-IT alignment models in the digital era: the impact of data inclusion // *Information Resources Management Journal*. 2022. Vol. 35, No. 1. P. 1–13. DOI: 10.4018/IRMJ.298972.

78. Sun Y., White J., Eade S., Schmidt D. C. ROAR: a QoS-oriented modeling framework for automated cloud resource allocation and optimization // *Journal of Systems and Software*. 2016. Vol. 116. P. 146–161. DOI: 10.1016/j.jss.2015.08.006.

79. Zubiaga A. Natural language processing in the era of large language models // *Frontiers in Artificial Intelligence*. 2024. Vol. 6. P. 1–5. DOI: 10.3389/frai.2023.1350306.

80. Nagavalli S. P., Tiwari S., Sarma W. Large language models and NLP: investigating challenges, opportunities, and the path to human-like language understanding // *International Research Journal of Engineering and Technology*. 2024. Vol. 12, No. 11. P. 571–584.

81. Lu P., Chen B., Liu S., Thapa R., Boen J., Zou J. OctoTools: An agentic framework with extensible tools for complex reasoning // *Proceedings of the ICLR 2025 Workshop on Foundation Models in the Wild*. 2025. DOI: 10.48550/arXiv.2502.11271.

82. Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., Goyal N., Küttler H., Lewis M., Yih W., Rocktäschel T., Kiela D., Riedel S. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks // *Advances in Neural Information Processing Systems*. 2020. Vol. 33. P. 9459–9474. DOI: 10.48550/arXiv.2005.11401.

83. Ishaya T. A service-oriented approach to business intelligence in telecoms industry // *Telematics and Informatics*. 2012. Vol. 29, No. 3. P. 273–285. DOI: 10.1016/j.tele.2012.01.004.

84. Wan J., Song S., Yu W., Liu Y., Cheng W., Huang F. et al. Omniparser: a unified framework for text spotting, key information extraction and table recognition // *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024. P. 15641–15653.

85. Mialon G., Dessì R., Lomeli M., Nalmpantis C., Pasunuru R., Raileanu R., Rozière B., Schick T., Dwivedi-Yu J., Celikyilmaz A., Grave E., LeCun Y., Scialom T. Augmented language models: a survey. 2023. DOI: 10.48550/arXiv.2302.07842.

86. Jiang Z., Xu F., Gao L., Sun Z., Liu Q., Dwivedi-Yu J., Yang Y., Callan J., Neubig G. Active retrieval augmented generation // *Proceedings of EMNLP 2023*. 2023. DOI: 10.18653/v1/2023.emnlp-main.495.

87. Shi W., Min S., Yasunaga M., Seo M., James R., Lewis M., Zettlemoyer L., Yih W. REPLUG: retrieval-augmented black-box language models // *Proceedings of NAACL-HLT 2024*. 2024. DOI: 10.18653/v1/2024.naacl-long.463.

88. Radford A., Kim J. W., Hallacy C., Ramesh A., Goh G., Agarwal S., Sastry G., Askell A., Mishkin P., Clark J., Krueger G., Sutskever I. Learning transferable visual models from natural language supervision // *Proceedings of the 38th International Conference on Machine Learning*. 2021. DOI: 10.48550/arXiv.2103.00020.

89. Liu H., Wang Z., Chen X., Li Z., Xiong F., Yu Q., Zhang W. HopRAG: multi-hop reasoning for logic-aware retrieval-augmented generation. 2025. DOI: 10.48550/arXiv.2502.12442.

90. Borgeaud S., Mensch A., Hoffmann J., Cai T., Rutherford E., Millican K., Sifre L. Improving language models by retrieving from trillions of tokens // *Proceedings of the International Conference on Machine Learning*. 2022. DOI: 10.48550/arXiv.2112.04426.

91. Schick T., Dwivedi-Yu J., Dessì R., Raileanu R., Lomeli M., Zettlemoyer L., Scialom T. Toolformer: language models can teach themselves to use tools. 2023. DOI: 10.48550/arXiv.2302.04761.

92. Chen W., Hu H., Chen X., Verga P., Cohen W. W. MuRAG: multimodal retrieval-augmented generator for open question answering over images and text // *Proceedings of EMNLP 2022*. 2022. P. 5558–5570. DOI: 10.18653/v1/2022.emnlp-main.375.

93. Caffagni D., Cocchi F., Moratelli N., Sarto S., Cornia M., Baraldi L., Cucchiara R. Wiki-LLaVA: hierarchical retrieval-augmented generation for multimodal LLMs // *CVPR Workshops*. 2024. DOI: 10.1109/CVPRW63382.2024.00188.

94. Yu S., Tang C., Xu B., Cui J., Ran J., Yan Y., Liu Z., Wang S., Han X., Liu Z., Sun M. VisRAG: vision-based retrieval-augmented generation on multi-modality documents. 2024. DOI: 10.48550/arXiv.2410.10594.

95. Cho J., Mahata D., Irsoy O., He Y., Bansal M. M3DocRAG: multi-modal retrieval is what you need for multi-page multi-document understanding. 2024. DOI: 10.48550/arXiv.2411.04952.

96. Suri M., Mathur P., Dernoncourt F., Gowswami K., Rossi R. A., Manocha D. VisDoM: multi-document QA with visually rich elements using multimodal retrieval-augmented generation. 2024. DOI: 10.48550/arXiv.2412.10704.

97. World Health Organization. *Global Tuberculosis Report 2024*. Geneva, 2024. [Електронний ресурс]. – Режим доступу: https://iris.who.int/ (дата звернення: 17.09.2025).

98. Kashyap P. React agents using LangChain // *Medium*. 2024. [Електронний ресурс]. – Режим доступу: https://medium.com/@piyushkashyap045/react-agents-using-langchain-388dab893fc9 (дата звернення: 17.09.2025).

99. Chroma. Chroma: the open-source AI application database. 2024. [Електронний ресурс]. – Режим доступу: https://www.trychroma.com/ (дата звернення: 17.09.2025).

100. OpenAI. New and improved embedding model: text-embedding-ada-002 // *OpenAI Blog*. 2022. [Електронний ресурс]. – Режим доступу: https://openai.com/index/new-and-improved-embedding-model/ (дата звернення: 17.09.2025).

101. OpenAI. GPT-4 technical report. 2023. DOI: 10.48550/arXiv.2303.08774.

102. PyMuPDF. Text extraction recipes [Електронний ресурс]. – Режим доступу: https://pymupdf.readthedocs.io/ (дата звернення: 17.09.2025).

103. Camelot. PDF table extraction for humans [Електронний ресурс]. – Режим доступу: https://camelot-py.readthedocs.io/ (дата звернення: 17.09.2025).

104. spaCy. Language processing pipelines [Електронний ресурс]. – Режим доступу: https://spacy.io/usage/processing-pipelines (дата звернення: 17.09.2025).

105. Reimers N., Gurevych I. Sentence-BERT: sentence embeddings using Siamese BERT-networks. 2019. DOI: 10.48550/arXiv.1908.10084.

106. Nowakowski G. Fuzzy queries on relational databases // *Proceedings of the International Interdisciplinary PhD Workshop*. 2018. P. 293–299. DOI: 10.1109/IIPHDW.2018.8388376.

107. Saraiva de Sousa N. F., Lachos Perez D. A., Rosa R. V., Santos M. A. S., Esteve Rothenberg C. Network service orchestration: a survey // *Computer Communications*. 2019. Vol. 142–143. P. 69–94. DOI: 10.1016/j.comcom.2019.04.008.

108. Manvi S. S., Krishna Shyam G. Resource management for infrastructure as a service in cloud computing: a survey // *Journal of Network and Computer Applications*. 2014. Vol. 41. P. 424–440. DOI: 10.1016/j.jnca.2013.10.004.

109. Widianto A., Subriadi A. P. IT service management evaluation method based on content, context and process approach: a literature review // *Procedia Computer Science*. 2022. Vol. 197. P. 410–419. DOI: 10.1016/j.procs.2021.12.157.

110. Yu Q., Zhao N., Li M., Li Z., Wang H., Zhang W., Sui K., Pei D. A survey on intelligent management of alerts and incidents in IT services // *Journal of Network and Computer Applications*. 2024. Vol. 224. Art. 103842. DOI: 10.1016/j.jnca.2024.103842.

111. Sun R., Gregor S. Reconceptualizing platforms in information systems research through the lens of service-dominant logic // *Journal of Strategic Information Systems*. 2023. Vol. 32, No. 3. Art. 101791. DOI: 10.1016/j.jsis.2023.101791.

112. Zakaria A. F., Lim S. C. J., Aamir M. A pricing optimization modelling for assisted decision making in telecommunication product-service bundling // *International Journal of Information Management Data Insights*. 2024. Vol. 4, No. 1. Art. 100212. DOI: 10.1016/j.jjimei.2024.100212.

113. Collins C., Dennehy D., Conboy K., Mikalef P. Artificial intelligence in information systems research: a systematic literature review and research agenda // *International Journal of Information Management*. 2021. Vol. 60. Art. 102383. DOI: 10.1016/j.ijinfomgt.2021.102383.

114. Khotin K., Shymkovych V., Kravets P., Novatsky A., Shymkovych L. Convolutional neural network for dog breed recognition system // *Адаптивні системи автоматичного управління*. 2024. No. 2(45). P. 3–14. DOI: 10.20535/1560-8956.45.2024.313022.

115. Steblianko O., Shymkovych V., Kravets P., Novatskyi A., Shymkovych L. Scientific article summarization model with unbounded input length // *Information, Computing and Intelligent Systems*. 2024. Vol. 5. P. 150–158. DOI: 10.20535/2786-8729.5.2024.314724.

116. Kobchenko V., Shymkovysh V., Kravets P., Novatskyi A., Shymkovysh L., Doroshenko A. An intelligent chatbot for evaluating the emotional

colouring of a message and responding accordingly // *Problems in Programming*. 2024. Vol. 1. P. 23–29. DOI: 10.15407/pp2024.01.23.

117. Ahlawat H., Aggarwal N., Gupta D. Automatic speech recognition: a survey of deep learning techniques and approaches // *International Journal of Cognitive Computing in Engineering*. 2025. Vol. 6. P. 201–237. DOI: 10.1016/j.ijcce.2024.12.007.

118. Yu J., Xu Y., Koh J. Y., Luong T., Baid G., Wang Z., Vasudevan V., Ku A., Yang Y., Ayan B. K., Hutchinson B., Han W., Parekh Z., Li X., Zhang H., Baldridge J., Wu Y. Scaling autoregressive models for content-rich text-to-image generation. 2022. DOI: 10.48550/arXiv.2206.10789.

119. Osypenko M., Shymkovych V., Kravets P., Novatsky A., Shymkovych L. Intelligent control system with reinforcement learning for solving video game tasks // *Адаптивні системи автоматичного управління*. 2024. No. 2(45). P. 34–46. DOI: 10.20535/1560-8956.45.2024.313065.

120. Kravets P., Novatskyi A., Shymkovych V., Rudakova A., Lebedenko Y., Rudakova H. Neural network model for laboratory stand control system controller with parallel mechanisms //*Advances in Computer Science for Engineering and Education VI : Proceedings of ICCSEEA 2023* /ed. by Z. Hu, I. Dychka, M. He. Cham : Springer, 2023. Vol. 181. P. 47–58. DOI: 10.1007/978-3-031-36118-0_5.

121. Wang X., He Z., Peng X. Artificial-intelligence-generated content with diffusion models: a literature review // *Mathematics*. 2024. Vol. 12, No. 7. Art. 977. DOI: 10.3390/math12070977.

122. Islam S., Elmekki H., Elsebai A., Bentahar J., Drawel N., Rjoub G., Pedrycz W. A comprehensive survey on applications of transformers for deep learning tasks // *Expert Systems with Applications*. 2023. Vol. 241. Art. 122666. DOI: 10.1016/j.eswa.2023.122666.

123. Le D. P. C., Wang D., Le V.-T. A comprehensive survey of recent transformers in image, video and diffusion models // *Computers, Materials & Continua*. 2024. Vol. 80, No. 1. P. 37–60. DOI: 10.32604/cmc.2024.050790.

124. Han K., Wang Y., Chen H., Chen X., Guo J., Liu Z., Tang Y., Xiao A., Xu C., Xu Y., Yang Z., Zhang Y., Tao D. A survey on vision transformer // *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2023. Vol. 45, No. 1. P. 87–110. DOI: 10.1109/TPAMI.2022.3152247.

125. Gavrilenko O., Zhurakovska O., Kohan A., Matviichuk R., Piskun A., Khavikova Y., Khalus O. The principle for forming a portfolio of public services based on the analysis of statistical information // *Eastern-European Journal of Enterprise Technologies*. 2022. No. 3(117). P. 57–64. DOI: 10.15587/1729-4061.2022.260136.

126. Gavrilenko O., Khomenko O., Zhurakovska O., Kohan A., Piskun A., Khalus O. Application of association rules for formation of public (administrative) services portfolio // *Advanced Information Systems*. 2022. Vol. 6, No. 4. P. 63–68. DOI: 10.20998/2522-9052.2022.4.09.

127. Gavrilenko O., Khomenko O., Zhurakovska O., Kogan A., Matviichuk R., Piskun A., Khavikova Y. Establishing the grouping principle of public services based on the analysis of similarity coefficients // *Eastern-European Journal of Enterprise Technologies*. 2023. No. 3(123). P. 22–29. DOI: 10.15587/1729-4061.2023.280218.

128. Гавриленко О., Жураковська О., Коган А., Богданова Н., Хоменко О. Аналіз впливу коефіцієнтів подібності на склад портфелів публічних сервісів // *Адаптивні системи автоматичного управління*. 2023. No. 1(42). P. 49–58. DOI: 10.20535/1560-8956.42.2023.279089.

129. Ester M., Kriegel H.-P., Sander J., Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise // *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*. AAAI Press, 1996. P. 226–231.

130. Kendall M. A new measure of rank correlation // *Biometrika*. 1938. Vol. 30, No. 1–2. P. 81–93. DOI: 10.1093/biomet/30.1-2.81.

131. Gavrilenko O., Zhurakovska O., Khomenko O. Analysis of the feasibility of implementing service portfolios // *Proceedings of the XXVIII*

*International Scientific and Practical Conference "Unusual Methods of Development of Science and Thoughts"*. Madrid, 2023. P. 182. [Електронний ресурс]. – Режим доступу: https://eu-conf.com/ua/events/unusual-methods-of-development-of-science-and-thoughts/ (дата звернення: 15.02.2026).

132. Gough J., Bryant D., Auburn M. *Mastering API Architecture: Design, Operate, and Evolve API-Based Systems*. Sebastopol : O'Reilly Media, 2022. ISBN 9781492088920.

133. Knote R., Janson A., Söllner M., Leimeister J. M.Value co-creation in smart services: a functional affordances perspective on smart personal assistants // *Journal of the Association for Information Systems*. 2020. Vol. 21, No. 2. P. 418–458. DOI: 10.17705/1jais.00580.

134. Holland J. H. Genetic algorithms // *Scientific American*. 1992. Vol. 267, No. 1. P. 66–73.

135. Goldberg D. E., Deb K. A comparative analysis of selection schemes used in genetic algorithms // *Foundations of Genetic Algorithms*. Morgan Kaufmann, 1991. P. 69–93.

136. Luke S., Spector L. A comparison of crossover and mutation in genetic programming // *Genetic Programming 1997 Proceedings*. 1997. P. 240–248.

137. Karaboga D., Gorkemli B., Ozturk C., Karaboga N. A comprehensive survey: artificial bee colony algorithm and applications // *Artificial Intelligence Review*. 2014. Vol. 42, No. 1. P. 21–57. DOI: 10.1007/s10462-012-9328-0.

138. Villani C. *Optimal Transport: Old and New*. Berlin : Springer, 2009. ISBN 9783540710492.

139. Загальна теорія статистики / за ред. А. В. Непрана, І. А. Дмитрієва. Харків : ПП Іванченка, 2022. 720 с.

140. Hastie T., Tibshirani R., Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York : Springer, 2009. ISBN 9780387848570.

141. Pele O., Werman M. Fast and robust Earth mover's distances // *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2009. P. 460–467. DOI: 10.1109/ICCV.2009.5459199.

142. Чимшир В. Інтелектуальний двигун і моделі та методи платформи комплексної підтримки процесів життєвого циклу інфокомунікаційних сервісів // *Технічні науки та технології*. 2025. No. 3(41). P. 284–295. DOI: 10.25140/2411-5363-2025-3(41)-284-295.

143. Yuvzhenko D., Chymshyr V., Shymkovych V., Znova K., Nowakowski G., Telenyk S. A multimodal retrieval-augmented generation system with ReAct agent logic for multi-hop reasoning // *Information, Computing and Intelligent Systems Journal*. 2025. No. 6. P. 42–57. DOI: 10.20535/2786-8729.6.2025.330777.

144. Шимкович В., Чимшир В., Знова К., Ювженко Д., Новаковський Г., Теленик С. Синтез архітектури сервісів на основі нейромережі з глибоким навчанням // *Адаптивні системи автоматичного управління*. 2025. Vol. 2, No. 47. P. 231–249. DOI: 10.20535/1560-8956.47.2025.340231.

# ДОДАТОК А. АКТ ВПРОВАДЖЕННЯ В НАВЧАЛЬНИЙ ПРОЦЕС

Декан факультету інформатики
та обчислювальної техніки
КПІ ім. Сікорського
_____ Ярослав КОРНАГА
«_» _____ 2025 року

**АКТ**

про впровадження в навчальний процес кафедри Інформаційних систем та технологій факультету Інформатики та обчислювальної техніки Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського» результатів дисертаційної роботи Чимшира Вячеслава Івановича «Моделі, методи та платформа підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг», поданої на здобуття наукового ступеня доктора філософії.

Ми, що нижче підписалися: завідувач кафедри Інформаційних систем та технологій, д.т.н. проф. Ролік О. І., вчений секретар кафедри, к.ф.-м.н., доц. Гавриленко О. В. підтверджуємо, що результати дисертаційної роботи Чимшира В. І. були включені в матеріали навчально-методичного забезпечення курсу «Проектування і реалізація інформаційних систем та технологій», а саме:

1) результати бізнес-аналізу, функціональні і нефункціональні вимоги, архітектура і програмна реалізація платформи підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг, які проєктуються, реалізуються, експлуатуються і надаються на основі моделі End-to-End;

2) велику мовну модель і RAG-систему, що забезпечують взаємодію компонентів платформи підтримки життєвого циклу сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг шляхом організації і підтримки виконання бізнес-процесів діяльності в умовах природномовної взаємодії з виконавцями з врахуванням проблемного підходу, накопичених текстових табличних і графічних даних;

3) мікросервісну архітектуру і програмну реалізацію системи формування портфелів сервісів, яка дозволяє використовувати сучасні методи формування портфелів сервісів для IT-компаній і їх провайдерів чи провайдерів і їх клієнтів, оцінювати і приймати рішення щодо портфелів, вибирати найбільш придатні методи за різними параметрами, редагувати рішення, дані осіб, які приймають рішення, переглядати статистики та тенденції щодо осіб, які приймають рішення, рішення;

4) нейромережу з глибоким навчанням, яка дозволяє будувати архітектуру сервісів в інформаційних системах провайдерів інформаційно-комунікаційних послуг, за яких досягається ефективне використання ресурсів, на основі параметрів сервісів, вимог провайдерів, характеристик їх клієнтів.

Впровадження результатів дисертаційної роботи Чимшира В. І. в навчальний процес дозволило підвищити якість підготовки студентів освітнього рівня «Бакалавр» 126 «Інформаційні системи та технології» на кафедрі Інформаційних систем та технологій КПІ ім. Ігоря Сікорського.

Завідувач кафедри
інформаційних систем та технологій,
д.т.н., проф. Олександр РОЛІК

Вчений секретар кафедри
інформаційних систем та технологій,
к.ф.-м.н., доц. Олена ГАВРИЛЕНКО

# ДОДАТОК Б. СХЕМИ АЛГОРИТМІВ РЕАЛІЗАЦІЇ ПЛАТФОРМИ ПІДТРИМКИ ЖИТТЄВОГО ЦИКЛУ СЕРВІСІВ

*Б.1 Алгоритм пошукової сесії*:

Крок 1. Формування запиту до векторної БД (query embedding).

Крок 2. Надсилання запиту паралельно до всіх трьох колекцій з параметром $k = 6$.

Крок 3. Сортування результатів за косинусною відстанню та об’єднання у спільну чергу пріоритетів (для опрацювання вибираються top-k = 10 документів з урахуванням емпірично визначеного відношення текст: таблиця: графіка. Виявилося, що саме вибір елементів цих колекцій у відношенні 6:2:2 забезпечує мінімальну довжину prompt-контексту без втрати Recall.

*Б.2 Експериментальне обґрунтування вибору кількості документів які повертає індекс для одного запиту*

Загалом параметр $k$ у контексті векторного пошуку означає кількість найближчих (за косинусною чи евклідовою відстанню) документів, які повертає індекс для одного запиту. Як зазначалося, у запропонованому рішенні запит надсилається одночасно до трьох колекцій, кожна з яких повертає $k$ результатів. Об’єднання і наступне ранжування результатів запитів забезпечує формування пулу з $3 * k$ кандидатів. Вибір Top-k з них для prompt-контексту здійснюється агентом.

Вибір найкращого значення параметру $k$ було здійснено емпіричним способом. У валідаційному тесті з 100 запитів значення $k$ змінювалося у діапазоні 3 – 8. Значення $k < 5$ знижували Recall, оскільки часто пропускалися релевантні фрагменти таблиць або капшенів. Значення $k > 7$ спричиняло зростання довжини prompt (токенів), що призводило до збільшенняи latency та вартість API-виклику GPT-4 без суттєвого приросту Accuracy.

Саме значення $k = 6$ супроводжувалося максимальним значенням F1-міри – гармонійного середнього Accuracy і Recall). Таким чином, кожна колекція повертає по 6 фрагментів, що призводить до формування фінального

пулу з 18 результатів. Вони надають додатковий контекст і саме до цих результатів застосовується відношення 6 : 2 : 2 тексту, таблиць і графіки.

Розрахунки показують, що при 18 кандидатах типового розміру (близько 120 токенів текст/таблиця і 80 токенів графіка) підсумковий контекст займає від 2 400 до 2 600 токенів. Якщо навіть врахувати власні Thought/Action-трейси ReAct-агента та відповіді моделі, робочий ліміт у 8 000 токенів не буде перевищений. Крім того, паралельний пошук у трьох колекціях із $k = 6$ (на Ryzen 5 + ChromaDB) вимагає середнього часу retrieval у діапазоні від близько 70 до 90 мс. Збільшення $k$ до 10 подовжує пошук на більш ніж 150 мс. Якщо врахувати ще й LLM-latency, відповідь буде значно повільнішою.

Отже, емпірично встановлено, що значення $k = 6$ забезпечує найкращий баланс між повнотою (Recall) і ефективністю (Latency / Token Cost) системи для обраного корпусу й апаратної конфігурації.

*Б.3 Схема процесу інтеграції RAG в платформу підтримки ЖЦ сервісів*

1. Вибір інструментів:

а) LangChain – для побудови ланцюжка RAG;

б) Pinecone – для векторного пошуку;

в) Sentence-BERT – для створення embeddings;

2. Побудова бази знань:

а) збір релевантних документів;

б) індексація даних у векторній базі;

3. Налаштування пошукового модуля:

а) оптимізація пошукових запитів;

б) використання метрик семантичної схожості для ранжування результатів;

4. Налаштування генеративного модуля:

а) використання техніки fine-tuning для адаптації LLM до виконання специфічних завдань (опціональний пункт, для стандартних моделей GPT-4, LLaMA 2, тонке налаштування моделі або fine-tuning не буде валідним);

б) використання Техніки Промпт Інжинірінгу;

в) інтеграція механізмів валідації результатів.

5. Тестування системи: а) перевірка точності відповіді з використанням метрики *Precision@k; б) оцінювання продуктивності та часу відповіді.

*Б.4 Процеси підготовки даних та генерації бази даних*

А. Вибір необхідних даних із PDF документа:

а) вилучення даних (PyMuPDF виокремлює текстові блоки, розмір яких перевищує 20 символів, Camelot розпізнає табличні контури й перетворює кожну таблицю на pandas.DataFrame);

б) очищення та токенізація текстових абзаців (виконується spaCy, для таблиць спочатку кожний рядок конкатенується через символ «|» для представлення двовимірних даних у лінійному текстовому форматі із збереженням послідовності колонок).

Б. Генерація векторів:

а) очищені абзаци й рядки подаються блоками по 128 елементів в OpenAI Embedding API (text-embedding-ada-002);

б) генерування моделлю 1536-вимірних семантичних векторів, у яких лексика, синтаксис і числові значення представлені у спільному векторному просторі (спільна embedding-модель для обох модальностей забезпечує узгодженість відстаней, що спрощує подальше ранжування під час пошуку).

В. Зберігання отриманих векторів і метаданих (id, page, modality) у ChromaDB: а) для тексту створюється колекція text_collection; б) для табличних рядків – колекція table_collection, кожна з індексом HNSW (параметри $M = 32$, efConstruction = 200). Такий поділ дозволяє оптимізувати швидкість nearest-neighbor пошуку в кожному підпросторі, зберігаючи при цьому можливість об’єднаного мультимодального ранжування результатів.

Д. Перенесення графічної інформації до векторної БД:

а) ідентифікація вбудованих зображень у PDF-файлі за допомогою функції page.get_image_list(full=True) з бібліотеки PyMuPDF (кожний рисунок отримує унікальний xref, координати bounding-box, номер сторінки);

б) із вказаної сторінки витягуються всі текстові блоки (page.get_text("blocks")) з їхніми координатами, що дає змогу проаналізувати, який саме підпис належить до конкретного зображення;

в) формування компактного, інформативного опису (власна функція HeuristicCaption v 0.2. відбирає ті текстові блоки, центр яких лежить у межах ± 120 px від верхньої або нижньої межі рисунка, ранжує їх за відстанню до самого зображення та об'єднує у рядок, обрізаний до 120 символів);

д) лінгвістична нормалізація отриманого капшена в spaCy (вилучення пунктуації та стоп-слова, кожне слово приводиться до леми, що підвищує семантичну узгодженість);

е) надсилання очищеного тексту в OpenAI Embedding API (text-embedding-ada-002);

є) перетворення тексту у 1536-вимірний вектор; ж) зберігання вектора з метаданими (id, page, modality = "image_caption") у колекції image_caption_collection у ChromaDB. Таким чином система репрезентує графічні об'єкти через їх текстові описи, що забезпечує швидкий семантичний пошук без обробки самих піксельних даних. Тобто, якщо підсумувати, Система не зберігає самі BMP/JPEG-файли й не будує CLIP-ембеддинги для піксельного вмісту. Замість цього вона індексує текстовий опис (caption) для кожного зображення.

*Б.5 Аналіз існуючих нейромереж для реалізації підходу до побудови архітектури сервісів*

Останнім часом згорткові нейронні мережі (CNN) стали центральним компонентом програм обробки зображень у міру розвитку нейронних мереж. Зорова система людини еволюціонувала до розрідженої та ефективної. Тому ми не обробляємо все наше поле зору тими ж ресурсами, на відміну від звивин. Наші очі виконують стратегію точки фіксації за допомогою системи зорової уваги, яка відіграє важливу роль у людському пізнанні. Натхненні системою уваги в людському зорі, обчислювальні механізми уваги були розроблені та інтегровані в нейронні мережі. Однією з цілей є зменшення обчислювального

навантаження, спричиненого операцією згортання, при цьому покращуючи продуктивність нейронної мережі.

Нейронні мережі на основі механізму уваги застосовувалися для різноманітних програм, таких як розпізнавання зображень або відстеження об’єктів. Нова архітектура нейронної мережі кодера-декодера на основі уваги була представлена для нейронного машинного перекладу (NMT) у 2015 році. Ідея цього підходу проілюстрована на рис. Б.1, де показано, як працюють механізми уваги в нейронних мережах. У цьому прикладі кодер приймає вхідне речення англійською мовою, а декодер виводить його переклад голландською мовою. Кодер та декодер включають рекурентні нейронні мережі (RNN). Кодер виводить приховані стани, а декодер приймає всі приховані стани як вхідні дані. Перед їх обробкою декодер застосовує механізм уваги, який оцінює кожен прихований стан. Потім він множить кожен прихований стан на його оцінку, до якої застосовується функція активації softmax. Зважені оцінки підсумовуються, і результат веде до контекстного вектора для декодера. Шляхом отримання зважених прихованих станів, які найбільш пов’язані з певними словами, декодер фокусується на відповідних частинах вхідних даних під час декодування.

Механізми уваги в нейронних мережах швидко прогресували, особливо для NMT. Одним з них є механізм самоуваги, який є основним будівельним блоком нейронних мереж трансформерів. Оригінальна нейронна мережа трансформер складається зі стеків кодерів-декодерів, які повністю базуються на механізмі самоуваги без будь-яких згорткових або рекурентних шарів. У кожному зі стеків кодера-декодера, які утворюють трансформер, є шість ідентичних шарів. Щоб проілюструвати модель, на рис. Б.1 показаний лише один стек кодера-декодера (тут ліворуч -трансформер з одним стеком кодера-декодера, у центрі – механізм багатоголової уваги, праворуч – механізм уваги скалярного добутку.

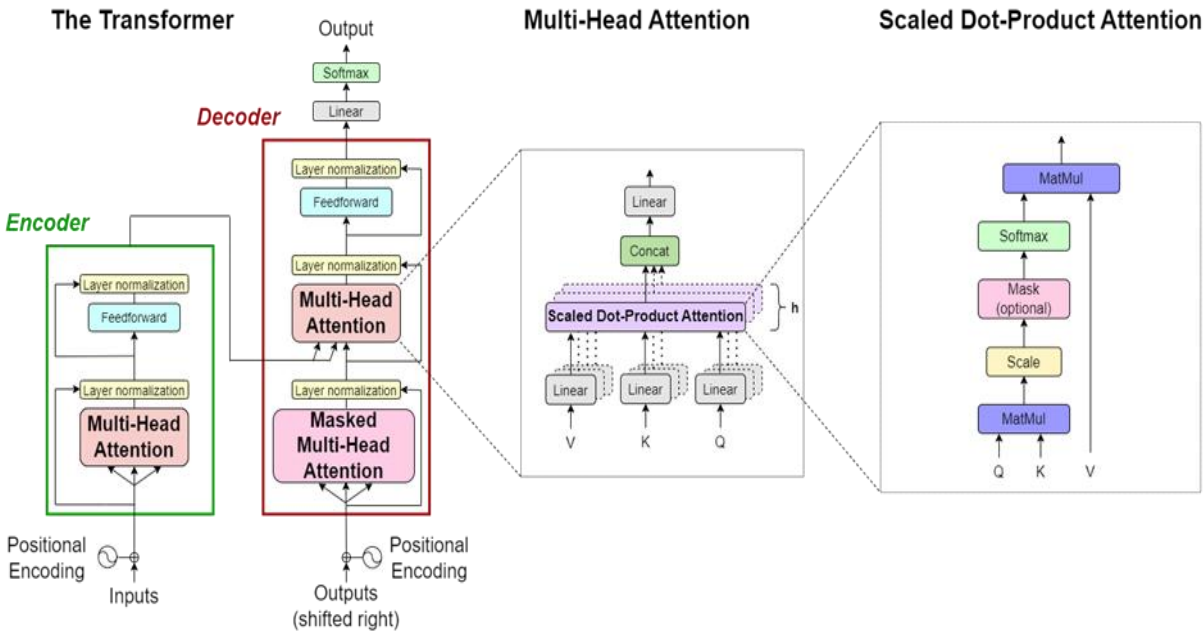


Рисунок Б.1 –Деталі архітектури нейронної мережі трансформер

Стеки кодера-декодера в трансформері складаються з повністю пов’язаних шарів і механізму багатоголової уваги, яка є свого роду механізмом самоконтролю. Механізм самоконтролю застосовує масштабований скалярний добуток уваги всередині себе. Як видно на рис. Б.1, ці механізми уваги використовують три вектори для кожного слова, а саме запит (Q), ключ (*K*) і значення (*V*). Ці вектори обчислюються шляхом множення вхідних даних на вагові матриці $W_q$, $W_k$ і $W_v$, які налаштовуються під час навчання. Загалом, кожне значення зважується функцією запиту з відповідним ключем. Вихідні дані обчислюються як зважена сума значень. Для обчислення скалярного добутку запиту з усіма ключами використовується функція softmax і формула Б.1 (тут кожен результат ділиться на квадратний корінь із розмірності ключів, щоб мати більш стабільні градієнти):

$$Attention(Q,K,V) = soft\max\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (Б.1)$$

Багатоголова увага розширює цю ідею, застосовуючи лінійні активації до вхідних даних (ключів, значень і запитів) h разів на основі різних вивчених лінійних представлень (рис. Б.1). Кожна з спроєктованих версій запитів, ключів і значень називається заголовками, для яких паралельно виконується

масштабований скалярний добуток. Отже, кілька разів з використанням різних наборів векторів запиту, ключів і значень обчислюється самоувага. Отже, інформацію на різних позиціях спільно обробляється, а на останньому кроці проекції з'єднуються. Щоб переконатися, що під час прогнозування наступного слова в реченні використовуються лише попередні вбудовані слова (токени), декодер застосовує замасковану увагу кількох голів.

Існують різні архітектури нейронних мереж трансформерів для різних завдань. Після значного прогресу в обробленні природної мови, трансформери були адаптовані до завдань оброблення зображень (приклад такої адаптаці наведений на рис. Б.2). Застосування самоконтролю у локальних околицях для кожного пікселя запиту добре сприяє створенню зображень і їх суперроздільності. Поточною сучасною моделлю є Vision Transformer (ViT) [103], який розбиває вхідне зображення на фрагменти перед тим, як Transformer візьме лінійне вбудовування цих фрагментів у послідовність як вхідні дані (рис. Б.2). ViT добре справляється із завданнями класифікації зображень, використовуючи при цьому менше обчислювальних ресурсів. Останні моделі, такі як ViT успішно вирішували завдання обробки зображень.

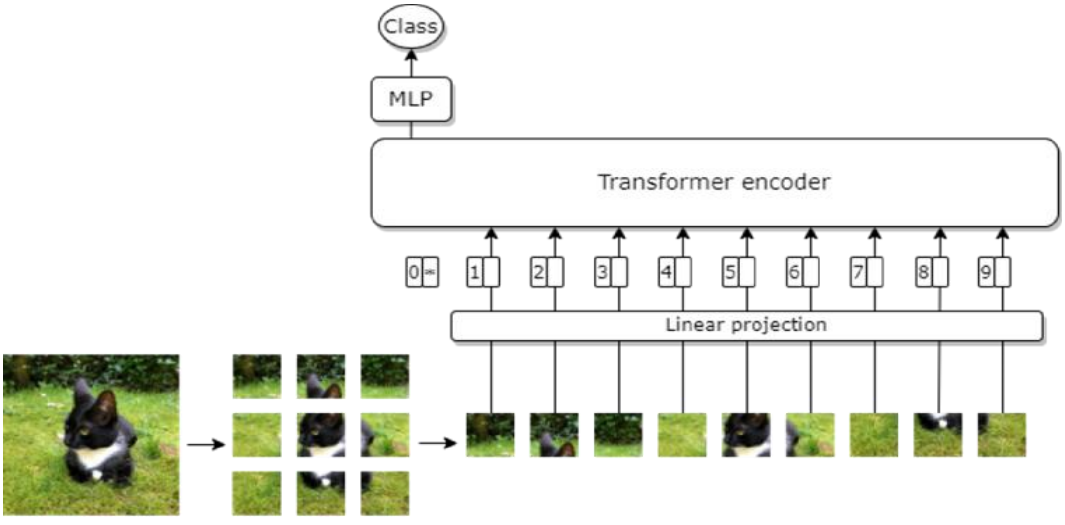


Рисунок Б.2 – Трансформер для обробки зображень

Сьогодні моделі перетворення тексту в зображення, такі як DALL-E 3 або Midjourney перетворили ШІ на популярний інструмент для створення захоплюючих зображень. Дифузійні моделі показали великий успіх у створенні

високоякісних зображень та перевершили GAN у синтезі зображень [100]. На відміну від GAN, навчання дифузійних моделей не потребує змагальності. Оригінальний метод шумозаглушення був натхненний нерівноважною термодинамікою, яка систематично руйнує структуру в розподілі даних, а потім відновлює дані. На основі цього методу ймовірнісні моделі дифузії з усуненням шуму були застосовані для створення зображень.

Дифузійні моделі потребують двох основних кроків у фазі навчання (рис. Б.3). На першому етапі, під час прямого (дифузійного) процесу, випадковий шум поступово додається до вхідного зображення, доки оригінальний вхід не стане повністю шумом. Це виконується фіксованим ланцюгом Маркова, який додає гаусівський шум для T послідовних кроків. На другому етапі, під час процесу реконструкції або зворотного процесу модель реконструює вихідні дані з шуму, отриманого в прямому процесі. Зворотний процес визначається як ланцюг Маркова з навченими переходами Гауса. Відповідно, передбачення щільності ймовірності в момент часу t залежить тільки від стану, досягнутого в момент часу ($t$-1). Тут $x_1,\ldots,x_T$ є латентами тієї ж розмірності, що й дані, які роблять моделі дифузії моделями латентної змінної.

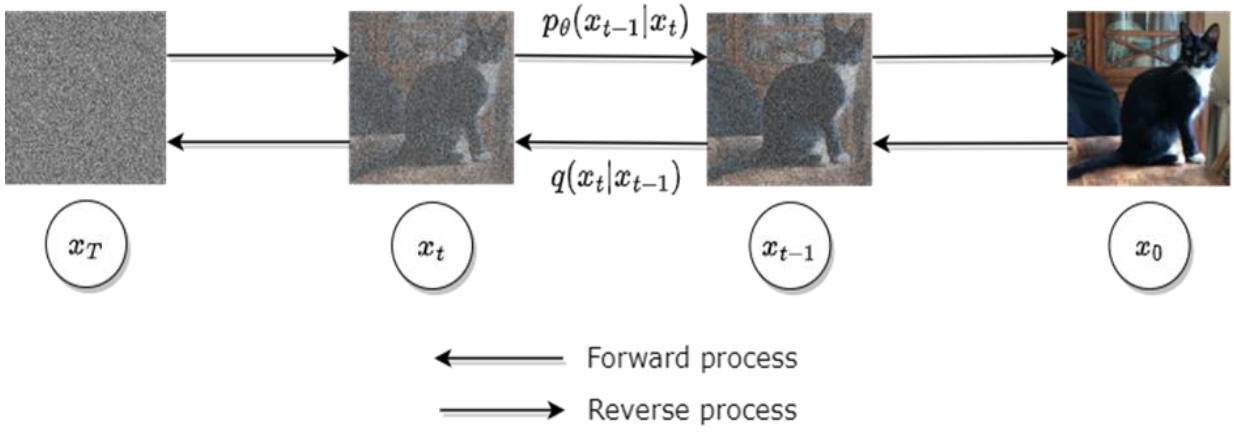


Рисунок Б.3 –Загальна дифузійної структура моделі

Загальна структура дифузійної моделі наведена на рис. Б.3. Оцінка щільності ймовірності на більш ранньому кроці часу, враховуючи поточний стан системи, є нетривіальною, тому зворотний процес вимагає навчання нейронної мережі. З цією метою всі попередні градієнти необхідні для

отримання необхідної оцінки. Для кожного кроку в ланцюзі Маркова нейронна мережа вчиться знімати шуми в зображенні. За бажанням, процес усунення шумів можна керувати текстом. У цьому випадку кодер Transformer відображає текстову підказку на токени, які потім передаються в нейронну мережу (рис. Б.4). Після навчання дифузійну модель можна використовувати для генерування даних шляхом простого проходження випадкового шуму (і додатково текстової підказки) через навчений процес усунення шумів. Слід зазначити, що цей процес потребує кількох кроків усунення шумів під час висновку, через що дифузійні моделі повільно генерують зображення, коли з’являється запит. У результаті було запропоновано кілька методів для прискорення висновку від багатоетапного до кількох і навіть однокрокового. Поточний метод SOTA для одноетапного та малоетапного висновку – Adversarial Diffusion Distillation, який використовує тренування змагальності, щоб підштовхнути генерацію зображень до розумних зображень із меншою кількістю кроків усунення шумів.

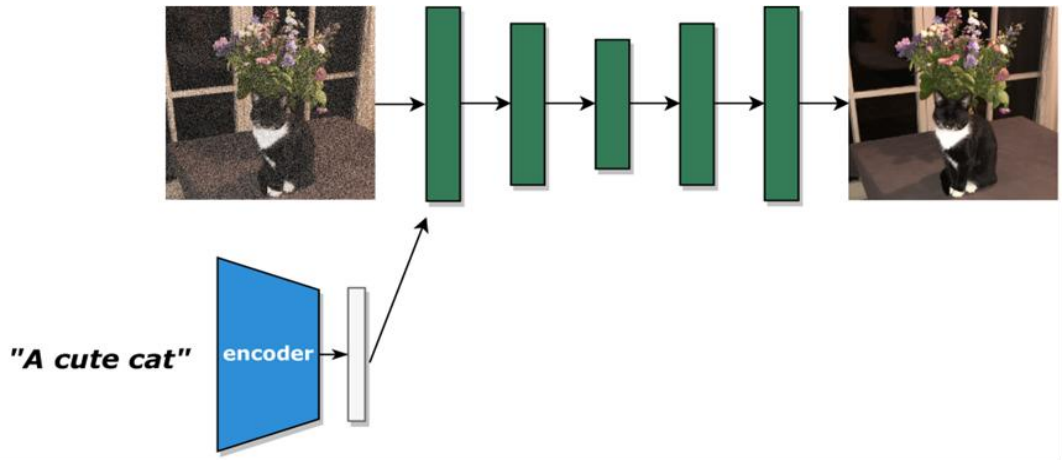


Рисунок Б.4 –Ілюстрація часового кроку роботи моделі

Глибока нейронна мережа вчиться перетворювати зашумлений вхідний сигнал у менш зашумлене зображення за допомогою текстової підказки, яка описує вміст зображення. Дифузійна модель нагадує VAE, які кодують вхідні дані як розподіл імовірностей, а потім вибірку з отриманого прихованого простору. Однак прямий процес відрізняє дифузійну модель від VAE, оскільки робить не потрібним навчання в цьому фіксованому ланцюгу Маркова.

# ДОДАТОК В. СХЕМИ АЛГОРИТМІВ ЗАПРОПОНОВАНИХ МЕТОДІВ ГРУПУВАННЯ СЕРВІСІВ У ПАКЕТИ

*Представлення накопичених даних і обчислення статистичних параметрів для реалізації статистичного методу*

Середню кількість клієнтів сервісу за період можна обчислити за (В.1):

$$\mathrm{W}_j(X_i) = \frac{\mathrm{m}_{ij}}{\mathrm{n}}, \quad \text{(В.1)}$$

де значенням параметру $m_{ij}$ є кількість клієнтів сервісу $S_i$ за період $j \in [1, n]$, а значенням параметру $n$ є встановлена користувачем кількість періодів.

Прибуток провайдера, отриманий від надання сервісу $S_i$ за період $j$, можна обчислити за допомогою відомої формули (В.2) [69]:

$$x_{ij} = m_{ij} * (y_i - z_i), \quad \text{(В.2)}$$

де $y_i$ – ціна сервісу $S_i$, $z_i$ – витрати на підтримку сервісу $S_i$.

Тепер на кроці 3 можна перейти до формування вибірок вибірок $X_i$ на основі результатів другого кроку для чергового сервісу, тобто на основі прибутків $x_{ij}$, $i \in [1, k]$, $j \in [1, n]$.

Передбачуваний прибуток провайдера за визначений період як вибіркове середнє для кожної вибірки $X_i$ можна обчислити за формулою (В.3):

$$\overline{X_\iota} = \sum_{j=1}^{n} x_{ij} * W_j(X_i). \quad \text{(В.3)}$$

Ризик можна обчислити як вибіркове середньоквадратичне відхилення за відомою формулою (В.4):

$$\sigma(\mathrm{X}_i) = \sqrt{\sum_{j=1}^{n} \mathrm{W}_j(\mathrm{X}_i) * \left(x_{ij} - \overline{X_\iota}\right)^2}. \quad \text{(В.4)}$$

Верхню і нижню межі передбачуваного прибутку можна обчислити за формулою (В.5):

$$[\overline{X_\iota} - 2 * \sigma(\mathrm{X}_i);\ \overline{X_\iota} + 2 * \sigma(\mathrm{X}_i)]. \quad \text{(В.5)}$$

Таблиця В.1 – Накопичені дані про надання сервісів по періодам

| Сервіси / Періоди | $S_1$ | $S_2$ | $S_3$ |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 1 | 5 | 3 | 4 |
| 2 | 3 | 6 | 2 |
| 3 | 2 | 4 | 5 |

Таблиця В.2 – Ціни за надання сервісів

| $S_i$ | $S_1$ | $S_2$ | $S_3$ |
|---|---|---|---|
| $y_i$ | 21 | 23 | 25 |

Таблиця В.3 – Затрати ІТ-компанії на надання сервісів

| $S_i$ | $S_1$ | $S_2$ | $S_3$ |
|---|---|---|---|
| $z_i$ | 7 | 8 | 9 |

Таблиця В.4 – Результати аналізу доцільності впровадження пакету сервісів

| | $S_1$ | $S_2$ | $S_3$ | Пакеть сервісів |
|---|---|---|---|---|
| Очікуваний прибуток за вказаний період | 53,2 | 48,23 | 57,27 | 158,7 |
| Ризик | 17,49 | 7,93 | 15,17 | 40,59 |
| Діапазон очікуваного прибутку | 18,22–88,18 | 32,37–64,09 | 26,93–87,61 | 77,52–239,88 |

*Форми для введення вхідних даних для алгоритму кластеризації*

Таблиця В.5 – Дані умов про надання сервісів

| | 1 | *j* | *k* |
|---|---|---|---|
| 1 | 1 | ... | 1 |
| *i* | ... | ... | ... |
| *n* | 1 | ... | 0 |

Таблиця В.6 – Дані про замовлення сервісів за період часу

| *No.* | 1 | 2 | 3 | 4 | 5 | 8 | 9 | 10 | 11 | 12 | 13 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 2 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 2 | 1 | 0 | 0 | 1 | 3 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 4 | 0 | 0 | 1 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 3 | 0 | 7 | 7 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 1 | 0 | 6 | 0 |
| 6 | 2 | 2 | 3 | 0 | 0 | 0 | 3 | 2 | 4 | 2 | 0 | 1 | 4 | 5 | 2 |
| 7 | 1 | 2 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 2 |

*Схема ймовірнісно-жадібного алгоритму розв'язання підзадачі етапу 2*

1 ***Вхід*** $k, m, \| d_{ij} \|, \| \beta_{ij} \|, \| r_{ij} \|, T, N$

2 ***Вихід*** $v_{best}$ // найкращий пакет сервісів (рекордний розв'язок)

3 $W_{best}$ // найкраще значення цільової функції (рекорд)

4 **Допоміжні величини**

5 $z_{ij},\ i = 1, \dots, k; j = 1, \dots, m$ // ознака дозволеності призначення $S_i - P_j$ (для яких на поточному кроці вистачає ресурсу)

6 **Обчислити** $\theta_{ij} := \frac{d_{ij}(1-r_{ij})}{\beta_{ij}}$ для всіх $i, j$ //«цінність» одиниці ресурсу

7 $W_{best} := 0$ // поточний рекорд

8 **for** $n := 1$ **to** $N$

9 $v_{ij} := 0$ для всіх $i, j$ // поточний пакет

10 $z_{ij} := 1$ для всіх $i, j$ // всі пари $i, j$ дозволені

11 $T_{used} := 0$ // використаний ресурс для поточного пакету $v$

12 **while** (є дозволені призначення) //для них ресурс $\beta_{ij} \le T - T_{used}$

14 Для дозволених пар $i, j$ **розрахувати** ймовірність вибору: $p_{ij} := \frac{\theta_{ij}}{\sum_{ij/z_{ij}=1} \theta_{ij}}$

15 Випадково **обрати** пару $i^*, j^*$ згідно $p_{ij}$

24 $v_{i^*j^*} := 1$

25 $z_{i^*j^*} := 0$

26 $T_{used} := T_{used} + \beta_{i^*j^*}$

27 для усіх непризначених пар $i, j$, яким не вистачить поточного ресурсу, **встановити** $z_{ij} := 0$

28 **endwhile**

29 **Обчислити** $W$ для отриманого пакету $v$

30 **if** $W > W_{best}$ **then** ($W_{best} := W$ та $v_{best} := v$)

31 **endfor**

32 Повернути $v_{best}$ та $W_{best}$

*Схема алгоритму мурашиної колонії для розв'язання підзадачі етапу 2*

1 ***Вхід*** $k, m, \| d_{ij} \|, \| \beta_{ij} \|, \| r_{ij} \|, T, A, \tau_0, \alpha, \beta, \rho, N$

2 ***Вихід*** $v_{best}$ , $W_{best}$

3 **Допоміжні величини**

4 $z_{ij},\ i = 1, \dots, k; j = 1, \dots, m$ // ознака дозволеності призначення $S_i - P_j$ (для яких на поточному кроці вистачає ресурсу)

5 $\tau$ // поточна матриця феромонів

6 **Обчислити** $\theta_{ij} := \frac{d_{ij}(1-r_{ij})}{\beta_{ij}}$ для всіх $i, j$ //«цінність» одиниці ресурсу для // призначення $S_i - P_j$

7 $W_{best} := 0$ // поточний рекорд

8 $\tau_{ij} := \tau_0$ для всіх $i, j$ // початковий феромон

9 **for** $n$ **from** 1 **to** $N$ не виконано умову зупинки (K

10 **for** $a$ **from** 1 **to** $A$ // цикл по мурахах
11 $v_{ij} \coloneqq 0$ для всіх $i,j$ // поточний пакет
12 $z_{ij} \coloneqq 1$ для всіх $i,j$ // всі пари $i,j$ дозволені
14 $T_{used} \coloneqq 0$ // використаний ресурс для поточного пакету $v$
15
24 **while** (існують дозволені пари) //для них ресурс $\beta_{ij} \le T - T_{used}$
25 Для кожної дозволеної пари $i,j$ **обчислити** імовірність вибору $p_{ij}$ за формулою (9)
26 Випадково **обрати** пару $i^*, j^*$ згідно $p_{ij}$
27 $v_{i^*j^*} \coloneqq 1$
28 $z_{i^*j^*} \coloneqq 0$
29 $T_{used} \coloneqq T_{used} + \beta_{i^*j^*}$
30 для усіх дозволених пар $i,j$, де $\beta_{ij} > T - T_{used}$, **встановити** $z_{ij} \coloneqq 0$
31 **endwhile**
32 **Обчислити** дохід $W$ для поточного пакету $v$
33 **if** $W > W_{best}$ **then** ($W_{best} \coloneqq W$ та $v_{best} \coloneqq v$)
34 **endfor**
35 **Оновити** матрицю феромонів за формулою (11)
36 **endfor**
37 Повернути $v_{best}$ та $W_{best}$

*Схема алгоритму знаходження компромісного розв'язку*

1 ***Вхід*** $k, m, T, v^1 = \{v_{ij}^1\}, v^2 = \{v_{ij}^2\}, \omega_1, \omega_2$
2 ***Вихід*** $v$ // компромісний пакет сервісів
3 $W, Q$ // значення цільових функцій
4 $v_{ij} \coloneqq 0$ для всіх $i,j$ // поточний пакет
5 $T_{used} \coloneqq 0$ // використаний ресурс для поточного пакету $v$
6 *// Додавання до пакету спільних для обох розв'язків призначень*
7 **for** $i \coloneqq 1$ **to** $k$ та $j \coloneqq 1$ **to** $m$
8 **if** ($v_{ij}^1 = 1$ і $v_{ij}^2 = 1$ і $T_{used} + \beta_{ij} \le T$) **then**
9 $v_{ij} \coloneqq 1$
10 $T_{used} \coloneqq T_{used} + \beta_{ij}$
11 **endif**
12 **endfor**
13 *// Обчисення компромісної цінності для решти призначень*
14 $C \coloneqq \emptyset$ //список претендентів на включення в пакет
15 **for** $i \coloneqq 1$ **to** $k$ та $j \coloneqq 1$ **to** $m$
16 **if** ($v_{ij}^1 + v_{ij}^2 = 1$ ) **then**
17 $$\theta_{ij} \coloneqq \frac{\omega_1 d_{ij}(1 - r_{ij}) + \omega_2 (p_{ij} - d_{ij}(1 - r_{ij}))}{\beta_{ij}}$$
18 $C \coloneqq C \cup (i, j, \theta_{ij})$
19 **endif**

```
endfor
Відсортувати список C за не зростанням θ_ij
// Додавання призначень до вичерпання ресурсу
for c := 1 to |C|
        if (T_used + β_ij ≤ T ) then
                v_ij := 1
                T_used := T_used + β_ij
        endif
endfor
Для портфеля v обчислити W та Q
Повернути v, W, Q
```

*Узагальнена схема модифікованого варіанту керованого генетичного алгоритму*

```
1   Вхід        q
                m
                C_lj,  l = 1, … , q; j = 1, … , m
                H_lj,  l = 1, … , q; j = 1, … , m
                B
2   Вихід       w_lj,  l = 1, … , q; j = 1, … , m
                y_lj,  l = 1, … , q; j = 1, … , m
                F
                Φ_j, j = 1, … , m
                // якщо p = 1,  то реалізується схрещення
3   p = 1       // якщо p = 0,  то реалізується мутація
4   w ← 0
5   while не виконана умова 2 do
6                   Створити початкову популяцію
7                   Визначити рекордний розв'язок
8                   while не виконана умова 1 do
9                           Обрати N/2 осіб
10                          if p = 1 then
11                                  Розбити обрані особини на пари
12                                  Схрестити обрані пари
13                          else
14                                  Мутувати кожну обрану особину
14                          endif
16                          Оновити популяцію
17                          Оновити рекордний розв'язок за необхідності
18                          Перерахувати p
19                  end while
20                  Зміна w
21                  Переоцінити особини популяції
22                  Оновити рекордний розв'язок за необхідності
23  end while
```

*Керований генетичний алгоритм*

Таблиця В.5

| № типу задач | Параметри задач/ Параметри КГА | Кількість згенерованих задач | %, коли найкращий результат отримано алгоритмом | |
|---|---|---|---|---|
| | | | КГА | ЕА |
| 1 | $m = 5, q = 5,$ $c_{ij} \in [1, 20], d_{ij} \in [50, 80],$ $B = 450/$ $N = 50$ | 50 | 98% | 2% |
| 2 | $m = 10, q = 10,$ $c_{ij} \in [1, 20], d_{ij} \in [50, 80],$ $B = 1350$ / $N = 100$ | 50 | 92% | 8% |
| 3 | $m = 15, q = 20,$ $c_{ij} \in [1, 20], d_{ij} \in [50, 80],$ $B = 2500$ / $N = 150$ | 20 | 6% | 94% |
| 4 | $m = 15, q = 20,$ $c_{ij} \in [1, 20], d_{ij} \in [21, 30],$ $B = 2500$ / $N = 150$ | 20 | 90% | 10% |

# ДОДАТОК Д. ПРИКЛАДИ КОДУВАННЯ ВХІДНИХ ДАНИХ І ВИМОГ ДО АРХІТЕКТУРИ

Вхід нейронної мережі побудови архітектур сервісів складається із значень характеристик, поділених на такі шість груп:

1. Загальні характеристики бізнес-діяльності провайдерів ІКП;
2. Кількісні характеристики сервісів, що надаються провайдером ІКП за підтримки ІТ-компанії, та клієнтів;
3. Функціональні вимоги до компонентів системи;
4. Нефункціональні вимоги до сервісів (параметри SLA та ін.);
5. Специфічні вимоги до сервісів, які надаються провайдером ІКП;
6. Додаткові вимоги провайдера ІКП до сервісів.

# Вимоги до архітектури в закодованому вигляді

X_data =

[0,1,0,0.5,0.8,0.8,0.9,0.3,0.7,0.7,0.8,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0.7,0.5,0,0,
1,0,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,0,0.7,0.7,0,0.5,0,0,0,0,0,0,0,0,0,0,0,0.7,0.7,0,1,0,0,1,0,
0,0,0,0,0,0,0,0,0.4,0.4,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0.5,0.5,0.5,0.3,0.9,0.6,0.99,0.89,0,0,
0.99,0.02,0.01,0,0.8,0.1,0.2,0.7,0.7,0,0.99,0.2,0.8,0.5,0.5,0,0,0,0.95,0.03,0.02,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0.99,0.99,1,1,1,1,1,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,
0,1,0.2,0.6,1,1,1,1,1,1,1,0.99,1,1,0.3,1,1,1,1,1,1,0,0,1,0.2,0.6,1,1,1,1,1,1,1,0.99,1,1,0.3,1,
1,1,1,1,1,0,0,1,0.2,0.6,1,1,1,1,1,1,1,0.99,1,1,0.3,1,1,1,1,1,1,0,0,1,0.2,0.6,1,1,1,1,1,1,1,0.99,
1,1,0.3,1,1,1,1,1,1,1,1,0,0,0,1,0,0,1,0,0,0,0,0,0,1,0,1,2,0,1,1,2,1,1,2,2,2,1,0,0,0,0,0,0,0,
0,1,1,0,0,0,1]

```
# Архітектура ситеми в закодованому вигляді
Y_data = [
    [0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0],
    [0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [1,0,0,1,2,1,2,2,1,0,0,0,0,1,1,0,0,0,0,0,0],
    [1,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0],
    [2,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0],
    [1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0],
    [1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0],
    [0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
]

#Перетворення вимог і архітектури в тензори
x = torch.tensor(X_data, dtype=torch.float32).unsqueeze(0)  # (1, 441)
y = torch.tensor(Y_data, dtype=torch.long).flatten().unsqueeze(0)  # (1, 424)
# Параметри нейромережевої моделі трансформер
input_seq_len = x.shape[1]          # 441
```

```
output_seq_len = y.shape[1]        # 424
d_model = 128                      #розмір embedding
nhead = 8                          #кількість голів уваги
num_layers = 3                     #шарів трансформера
vocab_size = 4                     #кількість класів для y (0,1,2, і 3 для паддінгу/пустого) – тут 3 цілочисельних класи, можна 4
```

Приклад коду вбудовування позицій:

```
# --- (positional encoding) ---
class PositionalEncoding(nn.Module):
    def __init__(self, d_model, max_len=5000):
        super().__init__()
        pe = torch.zeros(max_len, d_model)
        position = torch.arange(0, max_len).unsqueeze(1)
        div_term = torch.exp(torch.arange(0, d_model, 2) * (-torch.log(torch.tensor(10000.0)) / d_model))
        pe[:, 0::2] = torch.sin(position * div_term)
        pe[:, 1::2] = torch.cos(position * div_term)
        pe = pe.unsqueeze(0)  # (1, max_len, d_model)
        self.register_buffer('pe', pe)
    def forward(self, x):
        x = x + self.pe[:, :x.size(1)]
        return x
```

Приклад коду моделі:

```
class SimpleTransformer(nn.Module):
    def __init__(self):
        super().__init__()
        # Вбудовування входу (X_data – float) – через лінійний шар в d_model
```

```
        self.input_embed = nn.Linear(1, d_model)
        self.pos_encoder = PositionalEncoding(d_model)
        # Вбудовування для цілей – класів 0..3
        self.output_embed = nn.Embedding(vocab_size, d_model)
        self.pos_decoder = PositionalEncoding(d_model)
       # Трансформер
        self.transformer = nn.Transformer(d_model=d_model, nhead=nhead,
num_encoder_layers=num_layers, num_decoder_layers=num_layers)
        # Вихідний лінійний шар (класи)
        self.fc_out = nn.Linear(d_model, vocab_size)
    def forward(self, src, tgt):
        src = src.unsqueeze(-1)
        src = self.input_embed(src)
        src = self.pos_encoder(src)
        src = src.permute(1, 0, 2)
        tgt = self.output_embed(tgt)
        tgt = self.pos_decoder(tgt)
        tgt = tgt.permute(1, 0, 2)
        tgt_mask =
self.transformer.generate_square_subsequent_mask(tgt.size(0)).to(tgt.device)
        output = self.transformer(src, tgt, tgt_mask=tgt_mask)
        output = output.permute(1, 0, 2)
        return self.fc_out(output)
```

Дані кроку навчання нейронної мережі:

```
Epoch 50, Loss: 0.2485309541
Epoch 100, Loss: 0.1081207395
Epoch 150, Loss: 0.0339898467
Epoch 200, Loss: 0.0253857505
Epoch 250, Loss: 0.0081044007
Epoch 300, Loss: 0.0014549542
Epoch 350, Loss: 0.0011266612
Epoch 400, Loss: 0.0005256127
Epoch 450, Loss: 0.0054316204
Epoch 500, Loss: 0.0021697709
Epoch 550, Loss: 0.0004874875
Epoch 600, Loss: 0.0013051059
Epoch 650, Loss: 0.0002616614
Epoch 700, Loss: 0.0005597562
Epoch 750, Loss: 0.0013376080
Epoch 800, Loss: 0.0020588941
Epoch 850, Loss: 0.0064194687
Epoch 900, Loss: 0.0025966107
Epoch 950, Loss: 0.0169308409
Epoch 1000, Loss: 0.0003094980
Epoch 1050, Loss: 0.0180497523
Epoch 1100, Loss: 0.0112060383
```

```
Epoch 1150, Loss: 0.0002630538          Epoch 1950, Loss: 0.0000340328
Epoch 1200, Loss: 0.0007531480          Epoch 2000, Loss: 0.0000306932
Epoch 1250, Loss: 0.0001210206          Epoch 2050, Loss: 0.0000290106
Epoch 1300, Loss: 0.0001028961          Epoch 2100, Loss: 0.0000270610
Epoch 1350, Loss: 0.0008078362          Epoch 2150, Loss: 0.0000244728
Epoch 1400, Loss: 0.0001114520          Epoch 2200, Loss: 0.0000235164
Epoch 1450, Loss: 0.0000770486          Epoch 2250, Loss: 0.0000223041
Epoch 1500, Loss: 0.0000718153          Epoch 2300, Loss: 0.0000216388
Epoch 1550, Loss: 0.0000613159          Epoch 2350, Loss: 0.0000209564
Epoch 1600, Loss: 0.0001203779          Epoch 2400, Loss: 0.0000208680
Epoch 1650, Loss: 0.0000534833          Epoch 2450, Loss: 0.0000195671
Epoch 1700, Loss: 0.0000482199          Epoch 2500, Loss: 0.0000177165
Epoch 1750, Loss: 0.0000432101          Epoch 2550, Loss: 0.0000172189
Epoch 1800, Loss: 0.0000397460          Epoch 2600, Loss: 0.0000158058
Epoch 1850, Loss: 0.0000373399          Epoch 2650, Loss: 0.0000147961
Epoch 1900, Loss: 0.0000344137          Epoch 2700, Loss: 0.0000144547
```

Дані кроку тестування нейронної мережі після проведення навчання:

```
Передбачено (Y_data):
tensor([[0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0],
        [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 1, 2, 1, 2, 2, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
        [2, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])

Значення за умовою (Y_data):
tensor([[0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0],
        [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 1, 2, 1, 2, 2, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
        [2, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

# ДОДАТОК Е. ЛІСТИНГ КОДУ ПРОГРАМНОГО МОДУЛЮ ПРОТОТИПУ ВЕЛИКОЇ МОВНОЇ МОДЕЛІ


```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": 1,
   "metadata": {
    "ExecuteTime": {
     "end_time": "2025-07-10T14:50:09.457312500Z",
     "start_time": "2025-07-10T14:50:08.236304100Z"
    },
    "collapsed": false,
    "jupyter": {
     "outputs_hidden": false
    }
   },
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Requirement already satisfied: sentence-transformers in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (5.0.0)\n",
      "Requirement already satisfied: scikit-learn in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (1.7.0)\n",
      "Requirement already satisfied: pandas in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (2.3.0)\n",
      "Requirement already satisfied: numpy in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (2.2.6)\n",
      "Requirement already satisfied: requests in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (2.32.3)\n",
      "Requirement already satisfied: transformers<5.0.0,>=4.41.0 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
sentence-transformers) (4.53.0)\n",
      "Requirement already satisfied: tqdm in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
sentence-transformers) (4.67.1)\n",
      "Requirement already satisfied: torch>=1.11.0 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
sentence-transformers) (2.7.1+cu118)\n",
```

"Requirement already satisfied: scipy in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from sentence-transformers) (1.16.0)\n",
"Requirement already satisfied: huggingface-hub>=0.20.0 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from sentence-transformers) (0.32.0)\n",
"Requirement already satisfied: Pillow in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from sentence-transformers) (11.3.0)\n",
"Requirement already satisfied: typing_extensions>=4.5.0 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from sentence-transformers) (4.13.2)\n",
"Requirement already satisfied: filelock in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from transformers<5.0.0,>=4.41.0->sentence-transformers) (3.18.0)\n",
"Requirement already satisfied: packaging>=20.0 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from transformers<5.0.0,>=4.41.0->sentence-transformers) (24.2)\n",
"Requirement already satisfied: pyyaml>=5.1 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from transformers<5.0.0,>=4.41.0->sentence-transformers) (6.0.2)\n",
"Requirement already satisfied: regex!=2019.12.17 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from transformers<5.0.0,>=4.41.0->sentence-transformers) (2024.11.6)\n",
"Requirement already satisfied: tokenizers<0.22,>=0.21 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from transformers<5.0.0,>=4.41.0->sentence-transformers) (0.21.1)\n",
"Requirement already satisfied: safetensors>=0.4.3 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from transformers<5.0.0,>=4.41.0->sentence-transformers) (0.5.3)\n",
"Requirement already satisfied: fsspec>=2023.5.0 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from huggingface-hub>=0.20.0->sentence-transformers) (2025.3.0)\n",
"Requirement already satisfied: joblib>=1.2.0 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from scikit-learn) (1.5.1)\n",
"Requirement already satisfied: threadpoolctl>=3.1.0 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from scikit-learn) (3.6.0)\n",
"Requirement already satisfied: python-dateutil>=2.8.2 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from pandas) (2.9.0.post0)\n",
"Requirement already satisfied: pytz>=2020.1 in c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from pandas) (2025.2)\n",

```
            "Requirement already satisfied: tzdata>=2022.7 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
pandas) (2025.2)\n",
            "Requirement already satisfied: charset-normalizer<4,>=2 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
requests) (3.4.2)\n",
            "Requirement already satisfied: idna<4,>=2.5 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
requests) (3.10)\n",
            "Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
requests) (2.4.0)\n",
            "Requirement already satisfied: certifi>=2017.4.17 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
requests) (2025.4.26)\n",
            "Requirement already satisfied: six>=1.5 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
python-dateutil>=2.8.2->pandas) (1.17.0)\n",
            "Requirement already satisfied: sympy>=1.13.3 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
torch>=1.11.0->sentence-transformers) (1.14.0)\n",
            "Requirement already satisfied: networkx in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
torch>=1.11.0->sentence-transformers) (3.4.2)\n",
            "Requirement already satisfied: jinja2 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
torch>=1.11.0->sentence-transformers) (3.1.6)\n",
            "Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
sympy>=1.13.3->torch>=1.11.0->sentence-transformers) (1.3.0)\n",
            "Requirement already satisfied: colorama in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from tqdm-
>sentence-transformers) (0.4.6)\n",
            "Requirement already satisfied: MarkupSafe>=2.0 in
c:\\users\\max\\pycharmprojects\\llm_project\\venv\\lib\\site-packages (from
jinja2->torch>=1.11.0->sentence-transformers) (3.0.2)\n",
            "Note: you may need to restart the kernel to use updated packages.\n"
           ]
          }
         ],
         "source": [
          "pip install sentence-transformers scikit-learn pandas numpy requests"
         ]
        },
        {
```

```
"cell_type": "code",
"execution_count": null,
"metadata": {
 "ExecuteTime": {
  "start_time": "2025-07-09T15:09:50.616309300Z"
 },
 "collapsed": false,
 "is_executing": true,
 "jupyter": {
  "outputs_hidden": false
 }
},
"outputs": [
 {
  "name": "stdout",
  "output_type": "stream",
  "text": [
   "\n",
   "Docker LLM endpoint options:\n",
   "1. http://localhost:11434/v1/chat/completions (OpenAI-compatible)\n",
   "2. http://localhost:11434/generate (Simple)\n",
   "3. Custom endpoint\n"
  ]
 },
 {
  "name": "stdin",
  "output_type": "stream",
  "text": [
   "Choose endpoint (1-3) or press Enter for default:  2\n"
  ]
 },
 {
  "name": "stderr",
  "output_type": "stream",
  "text": [
   "INFO:__main__:Loaded CSV with 13506 rows and 21 columns\n",
   "INFO:__main__:Columns: ['Type', 'IBM Product', 'VRM', 'PID',
'Licence Type', 'MTM', 'GA', 'GA #', 'EOM', 'EOM #', 'EOD', 'EOD #', 'EOS', 'EOS
#', 'End All Support', 'Eligible service', 'Lifecycle Policy', 'Exception', 'Comments',
'More Info URL', 'Last modified']\n",
   "INFO:sentence_transformers.SentenceTransformer:Use pytorch
device_name: cuda:0\n",
   "INFO:sentence_transformers.SentenceTransformer:Load pretrained
SentenceTransformer: all-MiniLM-L6-v2\n"
  ]
```

```
},
{
 "name": "stdout",
 "output_type": "stream",
 "text": [
  "Loading CSV file: ibm_product_lifecycle_list.csv\n"
 ]
},
{
 "name": "stderr",
 "output_type": "stream",
 "text": [
  "INFO:__main__:Initialized embedding model: all-MiniLM-L6-v2\n",
  "INFO:__main__:Created knowledge base with 13506 entries\n"
 ]
},
{
 "data": {
  "application/vnd.jupyter.widget-view+json": {
   "model_id": "1931d5ae05934c78821509e7c7c45a44",
   "version_major": 2,
   "version_minor": 0
  },
  "text/plain": [
   "Batches:   0%|          | 0/423 [00:00<?, ?it/s]"
  ]
 },
 "metadata": {},
 "output_type": "display_data"
},
{
 "name": "stderr",
 "output_type": "stream",
 "text": [
  "INFO:__main__:Generated embeddings with shape: (13506, 384)\n"
 ]
},
{
 "name": "stdout",
 "output_type": "stream",
 "text": [
  "\n",
  "Loaded CSV Data Info:\n",
  "- Total rows: 13506\n",
```

```
            "- Columns: ['Type', 'IBM Product', 'VRM', 'PID', 'Licence Type',
'MTM', 'GA', 'GA #', 'EOM', 'EOM #', 'EOD', 'EOD #', 'EOS', 'EOS #', 'End All
Support', 'Eligible service', 'Lifecycle Policy', 'Exception', 'Comments', 'More Info
URL', 'Last modified']\n",
            "- Shape: (13506, 21)\n",
            "CSV ChatBot initialized! Type 'quit' to exit.\n",
            "Knowledge base loaded from: ibm_product_lifecycle_list.csv\n",
            "Using LLM endpoint: http://localhost:11434/generate\n",
            "-----------------------------------------------\n"
           ]
          },
          {
           "name": "stdin",
           "output_type": "stream",
           "text": [
            "\n",
            "You:  Show me the best IBM product by end date according to today's
date.\n"
           ]
          },
          {
           "name": "stdout",
           "output_type": "stream",
           "text": [
            "Bot: Thinking...\n"
           ]
          },
          {
           "data": {
            "application/vnd.jupyter.widget-view+json": {
             "model_id": "982120d2b61e4785bacfbc258db0c341",
             "version_major": 2,
             "version_minor": 0
            },
            "text/plain": [
             "Batches:   0%|          | 0/1 [00:00<?, ?it/s]"
            ]
           },
           "metadata": {},
           "output_type": "display_data"
          },
          {
           "name": "stdout",
           "output_type": "stream",
           "text": [
```

"Bot: To determine the best IBM product by end date according to today's date, I need to know today's date. Since I don't have real-time access to the current date, I'll assume a hypothetical date for demonstration purposes.\n",
"\n",
"Let's assume today's date is March 1, 2023.\n",
"\n",
"Based on the provided CSV data:\n",
"\n",
"Entry 1:\n",
"EOS: 2019-09-30\n",
"End All Support: 2021-09-30\n",
"\n",
"Entry 2:\n",
"EOS: 2020-04-30\n",
"End All Support: 2023-04-30 (which is after today's assumed date)\n",
"\n",
"Entry 3:\n",
"EOS: 2019-09-30\n",
"End All Support: 2022-09-30 (which is before today's assumed date)\n",
"\n",
"Since Entry 2 has an End All Support date that is later than today's assumed date, and no other entries have an EOS date after today's assumed date, Entry 2 appears to be the best IBM product by end date according to this hypothetical date.\n",
"\n",
"However, please note that without knowing the actual current date, this answer may not be accurate. If you provide the current date or a range of dates, I can give you a more precise answer based on the provided CSV data.\n"
]
},
{
"name": "stdin",
"output_type": "stream",
"text": [
"\n",
"You:  Provide names for these products.\n"
]
},
{
"name": "stdout",
"output_type": "stream",
"text": [
"Bot: Thinking...\n"
]
},

```
{
 "data": {
  "application/vnd.jupyter.widget-view+json": {
   "model_id": "02efa24170564a06ab152e275effacc9",
   "version_major": 2,
   "version_minor": 0
  },
  "text/plain": [
   "Batches:   0%|          | 0/1 [00:00<?, ?it/s]"
  ]
 },
 "metadata": {},
 "output_type": "display_data"
},
{
 "name": "stdout",
 "output_type": "stream",
 "text": [
  "Bot: Based on the provided CSV data, I can provide the product names as follows:\n",
  "\n",
  "1. Rational SoDA for Word\n",
  "2. Rational SoDA\n",
  "3. IBM Maximo Asset Health Insights\n"
 ]
},
{
 "name": "stdin",
 "output_type": "stream",
 "text": [
  "\n",
  "You:  Lifecycle policy for the first IBM product by date.\n"
 ]
},
{
 "name": "stdout",
 "output_type": "stream",
 "text": [
  "Bot: Thinking...\n"
 ]
},
{
 "data": {
  "application/vnd.jupyter.widget-view+json": {
   "model_id": "5967d839f6ff409dbd41686c5aae8f88",
```

```
          "version_major": 2,
          "version_minor": 0
         },
         "text/plain": [
          "Batches:   0%|          | 0/1 [00:00<?, ?it/s]"
         ]
        },
        "metadata": {},
        "output_type": "display_data"
       },
       {
        "name": "stdout",
        "output_type": "stream",
        "text": [
         "Bot: Based on the provided CSV data, I was unable to find any information about a lifecycle policy that is ordered chronologically (i.e., earliest End of Support date first). The Lifecycle Policy is mentioned for each product, but it does not indicate a specific ordering by date.\n"
        ]
       }
      ],
      "source": [
       "import pandas as pd\n",
       "import numpy as np\n",
       "from sentence_transformers import SentenceTransformer\n",
       "from sklearn.metrics.pairwise import cosine_similarity\n",
       "import requests\n",
       "import json\n",
       "import os\n",
       "from typing import List, Dict, Any\n",
       "import logging\n",
       "\n",
       "# Configure logging\n",
       "logging.basicConfig(level=logging.INFO)\n",
       "logger = logging.getLogger(__name__)\n",
       "\n",
       "class CSVChatBot:\n",
       "    def __init__(self, csv_file_path: str = \"ibm_product_lifecycle_list.csv\", llm_endpoint: str = \"http://localhost:11434/v1/chat/completions\", \n",
       "                 embedding_model: str = \"all-MiniLM-L6-v2\"):\n",
       "        \"\"\"\n",
       "        Initialize the CSV ChatBot\n",
       "        \n",
       "        Args:\n",
```

```
            "            csv_file_path: Path to the CSV file containing knowledge
base\n",
            "            llm_endpoint: API endpoint for the LLM (Docker container)\n",
            "            embedding_model: Name of the sentence transformer model for
embeddings\n",
            "        \"\"\"\n",
            "        self.csv_file_path = csv_file_path\n",
            "        self.llm_endpoint = llm_endpoint\n",
            "        self.embedding_model_name = embedding_model\n",
            "        \n",
            "        # Initialize components\n",
            "        self.df = None\n",
            "        self.embeddings = None\n",
            "        self.embedding_model = None\n",
            "        self.knowledge_base = []\n",
            "        \n",
            "        # Load and process data\n",
            "        self.load_csv_data()\n",
            "        self.initialize_embedding_model()\n",
            "        self.create_knowledge_base()\n",
            "        self.generate_embeddings()\n",
            "        \n",
            "    def load_csv_data(self):\n",
            "        \"\"\"Load CSV data into pandas DataFrame\"\"\"\n",
            "        try:\n",
            "            self.df = pd.read_csv(self.csv_file_path)\n",
            "            logger.info(f\"Loaded CSV with {len(self.df)} rows and
{len(self.df.columns)} columns\")\n",
            "            logger.info(f\"Columns: {list(self.df.columns)}\")\n",
            "        except Exception as e:\n",
            "            logger.error(f\"Error loading CSV file: {e}\")\n",
            "            raise\n",
            "    \n",
            "    def initialize_embedding_model(self):\n",
            "        \"\"\"Initialize the sentence transformer model for
embeddings\"\"\"\n",
            "        try:\n",
            "            self.embedding_model =
SentenceTransformer(self.embedding_model_name)\n",
            "            logger.info(f\"Initialized embedding model:
{self.embedding_model_name}\")\n",
            "        except Exception as e:\n",
            "            logger.error(f\"Error initializing embedding model: {e}\")\n",
            "            raise\n",
            "    \n",
```

```
            "    def create_knowledge_base(self):\n",
            "        \"\"\"Create knowledge base from CSV data\"\"\"\n",
            "        self.knowledge_base = []\n",
            "        \n",
            "        for idx, row in self.df.iterrows():\n",
            "            # Create a text representation of each row\n",
            "            row_text = \"\"\n",
            "            for col in self.df.columns:\n",
            "                value = row[col]\n",
            "                if pd.notna(value):  # Only include non-null values\n",
            "                    row_text += f\"{col}: {value}. \"\n",
            "            \n",
            "            self.knowledge_base.append({\n",
            "                'id': idx,\n",
            "                'text': row_text.strip(),\n",
            "                'data': row.to_dict()\n",
            "            })\n",
            "        \n",
            "        logger.info(f\"Created knowledge base with 
{len(self.knowledge_base)} entries\")\n",
            "    \n",
            "    def generate_embeddings(self):\n",
            "        \"\"\"Generate embeddings for all knowledge base entries\"\"\"\n",
            "        texts = [entry['text'] for entry in self.knowledge_base]\n",
            "        self.embeddings = self.embedding_model.encode(texts)\n",
            "        logger.info(f\"Generated embeddings with shape: 
{self.embeddings.shape}\")\n",
            "    \n",
            "    def find_relevant_context(self, query: str, top_k: int = 3) -> 
List[Dict]:\n",
            "        \"\"\"\n",
            "        Find the most relevant context from the knowledge base\n",
            "        \n",
            "        Args:\n",
            "            query: User query\n",
            "            top_k: Number of top results to return\n",
            "            \n",
            "        Returns:\n",
            "            List of relevant knowledge base entries\n",
            "        \"\"\"\n",
            "        # Generate embedding for the query\n",
            "        query_embedding = self.embedding_model.encode([query])\n",
            "        \n",
            "        # Calculate cosine similarity\n",
```

```
            "        similarities = cosine_similarity(query_embedding,
self.embeddings)[0]\n",
            "        \n",
            "        # Get top-k most similar entries\n",
            "        top_indices = np.argsort(similarities)[-top_k:][::-1]\n",
            "        \n",
            "        relevant_entries = []\n",
            "        for idx in top_indices:\n",
            "            relevant_entries.append({\n",
            "                'entry': self.knowledge_base[idx],\n",
            "                'similarity': similarities[idx]\n",
            "            })\n",
            "        \n",
            "        return relevant_entries\n",
            "    \n",
            "    def call_local_llm(self, prompt: str) -> str:\n",
            "        \"\"\"\n",
            "        Call local LLM via Docker container API\n",
            "        \n",
            "        Args:\n",
            "            prompt: The prompt to send to the LLM\n",
            "            \n",
            "        Returns:\n",
            "            LLM response\n",
            "        \"\"\"\n",
            "        try:\n",
            "            # Try OpenAI-compatible API format first\n",
            "            payload = {\n",
            "                \"model\": \"llama2\",\n",
            "                \"messages\": [\n",
            "                    {\"role\": \"user\", \"content\": prompt}\n",
            "                ],\n",
            "                \"temperature\": 0.7,\n",
            "                \"max_tokens\": 1000\n",
            "            }\n",
            "            \n",
            "            headers = {\n",
            "                \"Content-Type\": \"application/json\"\n",
            "            }\n",
            "            \n",
            "            response = requests.post(self.llm_endpoint, json=payload,
headers=headers)\n",
            "            \n",
            "            if response.status_code == 200:\n",
            "                result = response.json()\n",
```

```
            "                # Try to extract response from OpenAI format\n",
            "                if 'choices' in result and len(result['choices']) > 0:\n",
            "                    return result['choices'][0]['message']['content']\n",
            "                elif 'response' in result:\n",
            "                    return result['response']\n",
            "                else:\n",
            "                    return str(result)\n",
            "            else:\n",
            "                # Try alternative endpoint formats\n",
            "                alternative_endpoints = [\n",
            "                    \"http://localhost:11434/generate\",\n",
            "                    \"http://localhost:11434/api/generate\",\n",
            "                    \"http://localhost:8000/v1/chat/completions\",\n",
            "                    \"http://localhost:8000/generate\",\n",
            "                    \"http://localhost:8000/api/generate\"\n",
            "                ]\n",
            "                \n",
            "                for endpoint in alternative_endpoints:\n",
            "                    try:\n",
            "                        # Try Ollama-style format\n",
            "                        simple_payload = {\n",
            "                            \"model\": \"llama3.2\",\n",
            "                            \"prompt\": prompt,\n",
            "                            \"stream\": False\n",
            "                        }\n",
            "                        \n",
            "                        alt_response = requests.post(endpoint,
json=simple_payload)\n",
            "                        if alt_response.status_code == 200:\n",
            "                            alt_result = alt_response.json()\n",
            "                            if 'response' in alt_result:\n",
            "                                return alt_result['response']\n",
            "                    except:\n",
            "                        continue\n",
            "                \n",
            "                logger.error(f\"LLM API error: {response.status_code} –
{response.text}\")\n",
            "                return \"Error: Could not get response from local LLM\"\n",
            "                \n",
            "            except Exception as e:\n",
            "                logger.error(f\"Error calling local LLM: {e}\")\n",
            "                return \"Error: Could not connect to local LLM\"\n",
            "    \n",
            "    def generate_response(self, query: str) -> str:\n",
            "        \"\"\"\n",
```

```
            "        Generate response based on query and CSV data\n",
            "        \n",
            "        Args:\n",
            "            query: User query\n",
            "            \n",
            "        Returns:\n",
            "            Generated response\n",
            "        \"\"\"\n",
            "        # Find relevant context\n",
            "        relevant_entries = self.find_relevant_context(query)\n",
            "        \n",
            "        # Build context string\n",
            "        context = \"Based on the following information from the CSV
data:\\n\\n\"\n",
            "        for i, entry in enumerate(relevant_entries, 1):\n",
            "            context += f\"Entry {i} (similarity:
{entry['similarity']:.3f}):\\n\"\n",
            "            context += f\"{entry['entry']['text']}\\n\\n\"\n",
            "        \n",
            "        # Create prompt for LLM\n",
            "        prompt = f\"\"\"You are a helpful assistant that answers questions
based on provided CSV data.\n",
            "\n",
            "{context}\n",
            "\n",
            "User Question: {query}\n",
            "\n",
            "Please provide a helpful and accurate answer based on the CSV data
provided above. If the information is not available in the data, please say so
clearly.\n",
            "\n",
            "Answer:\"\"\"\n",
            "        \n",
            "        # Get response from local LLM\n",
            "        response = self.call_local_llm(prompt)\n",
            "        \n",
            "        return response\n",
            "    \n",
            "    def chat(self):\n",
            "        \"\"\"Interactive chat interface\"\"\"\n",
            "        print(\"CSV ChatBot initialized! Type 'quit' to exit.\")\n",
            "        print(f\"Knowledge base loaded from: {self.csv_file_path}\")\n",
            "        print(f\"Using LLM endpoint: {self.llm_endpoint}\")\n",
            "        print(\"-\" * 50)\n",
            "        \n",
```

```
            "        while True:\n",
            "            user_input = input(\"\\nYou: \").strip()\n",
            "            \n",
            "            if user_input.lower() in ['quit', 'exit', 'q']:\n",
            "                print(\"Goodbye!\")\n",
            "                break\n",
            "            \n",
            "            if not user_input:\n",
            "                continue\n",
            "            \n",
            "            print(\"Bot: Thinking...\")\n",
            "            response = self.generate_response(user_input)\n",
            "            print(f\"Bot: {response}\")\n",
            "    \n",
            "    def get_data_info(self) -> Dict[str, Any]:\n",
            "        \"\"\"Get information about the loaded CSV data\"\"\"\n",
            "        return {\n",
            "            'total_rows': len(self.df),\n",
            "            'columns': list(self.df.columns),\n",
            "            'shape': self.df.shape,\n",
            "            'data_types': self.df.dtypes.to_dict()\n",
            "        }\n",
            "\n",
            "# Example usage and setup\n",
            "if __name__ == \"__main__\":\n",
            "    # Get CSV file path from user\n",
            "    csv_file_path = \"ibm_product_lifecycle_list.csv\"\n",
            "    \n",
            "    # Check if file exists\n",
            "    if not os.path.exists(csv_file_path):\n",
            "        print(f\"Error: File '{csv_file_path}' not found!\")\n",
            "        print(\"Please provide a valid CSV file path.\")\n",
            "        exit(1)\n",
            "    \n",
            "    # Get LLM endpoint from user\n",
            "    print(\"\\nDocker LLM endpoint options:\")\n",
            "    print(\"1. http://localhost:11434/v1/chat/completions (OpenAI-
compatible)\")\n",
            "    print(\"2. http://localhost:11434/generate (Simple)\")\n",
            "    print(\"3. Custom endpoint\")\n",
            "    \n",
            "    endpoint_choice = input(\"Choose endpoint (1-3) or press Enter for
default: \").strip()\n",
            "    \n",
            "    if endpoint_choice == \"2\":\n",
```

```
            "        llm_endpoint = \"http://localhost:11434/generate\"\n",
            "    elif endpoint_choice == \"3\":\n",
            "        llm_endpoint = input(\"Enter custom endpoint: \").strip()\n",
            "    else:\n",
            "        llm_endpoint = \"http://localhost:11434/v1/chat/completions\"\n",
            "    \n",
            "    # Initialize and run chatbot\n",
            "    try:\n",
            "        print(f\"Loading CSV file: {csv_file_path}\")\n",
            "        bot = CSVChatBot(csv_file_path, llm_endpoint=llm_endpoint)\n",
            "        \n",
            "        # Print data info\n",
            "        data_info = bot.get_data_info()\n",
            "        print(f\"\\nLoaded CSV Data Info:\")\n",
            "        print(f\"- Total rows: {data_info['total_rows']}\")\n",
            "        print(f\"- Columns: {data_info['columns']}\")\n",
            "        print(f\"- Shape: {data_info['shape']}\")\n",
            "        \n",
            "        # Start chat\n",
            "        bot.chat()\n",
            "        \n",
            "    except Exception as e:\n",
            "        print(f\"Error: {e}\")\n",
            "        print(\"\\nSetup Instructions:\")\n",
            "        print(\"1. Make sure your Docker container is running:\")\n",
            "        print(\"   docker run -p 11434:11434 --gpus all
alpine/llama3.2:latest\")\n",
            "        print(\"2. Install required packages:\")\n",
            "        print(\"   pip install sentence-transformers scikit-learn pandas
numpy requests\")\n",
            "        print(\"3. Ensure your CSV file exists and is readable\")\n",
            "        print(\"4. Check that the Docker container is accessible at the
specified endpoint\")"
           ]
          },
          {
           "cell_type": "code",
           "execution_count": null,
           "metadata": {
            "collapsed": false,
            "is_executing": true,
            "jupyter": {
             "outputs_hidden": false
            }
           },
```

```
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "accelerator": "GPU",
  "colab": {
   "gpuType": "T4",
   "provenance": []
  },
  "kernelspec": {
   "display_name": "Python 3 (ipykernel)",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.11.7"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 4
}
```

# ДОДАТОК Ж. ЛІСТИНГ КОДУ МІКРОСЕРВІСА КОМБІНОВАНОГО МЕТОДУ

```
# import numpy as np
# import pandas as pd
# import io
# from fastapi import FastAPI, UploadFile, File, HTTPException
# from pydantic import BaseModel
# from typing import List, Dict, Any, Optional
# import copy
# import random
#
#
# class CalculationRequest(BaseModel):
#     price: List[float]
#     support_cost: List[float]
#     daily_orders: List[List[float]]
#     n_days: int
#
#
# class Particle:
#     def __init__(self, k, m, bounds=(0, 1)):
#         # Position (solution) represented as a k x m matrix
#         self.position = np.random.uniform(bounds[0], bounds[1], (k, m))
#         # Velocity of the particle
#         self.velocity = np.random.uniform(-0.1, 0.1, (k, m))
#         # Personal best position
#         self.pbest_position = self.position.copy()
#                 # Personal best objective values [IT_company_objective, providers_objective]
#         self.pbest_objectives = [float('-inf'), float('-inf')]
#         # Current objective values
#         self.current_objectives = [float('-inf'), float('-inf')]
#         # Dominance rank (for crowding)
#         self.rank = 0
#         # Crowding distance
#         self.crowding_distance = 0
#
#     def update_velocity(self, global_best_position, w=0.7, c1=1.5, c2=1.5):
#       r1 = np.random.random((self.position.shape[0], self.position.shape[1]))
#       r2 = np.random.random((self.position.shape[0], self.position.shape[1]))
#
#         cognitive_component = c1 * r1 * (self.pbest_position – self.position)
#         social_component = c2 * r2 * (global_best_position – self.position)
```

```
        #
        #               self.velocity = w * self.velocity + cognitive_component +
social_component
        #
        #       # Limit velocity to avoid large jumps
        #       self.velocity = np.clip(self.velocity, -0.1, 0.1)
        #
        #    def update_position(self, bounds=(0, 1)):
        #       """Update position based on velocity"""
        #       self.position = self.position + self.velocity
        #       # Keep position within bounds
        #       self.position = np.clip(self.position, bounds[0], bounds[1])
        #
        #
        # class MOPSO:
        #              def __init__(self, calculation_request, discount_rates=None,
num_providers=3,
        #               num_particles=50, max_iter=100, bounds=(0, 1)):
        #
        #       # Extract data from calculation request
        #       self.price = np.array(calculation_request.price)
        #       self.support_cost = np.array(calculation_request.support_cost)
        #       self.daily_orders = np.array(calculation_request.daily_orders)
        #       self.n_days = calculation_request.n_days
        #
        #       # Dimensions
        #       self.k = len(self.price)  # Number of services
        #       self.m = num_providers  # Number of providers
        #
        #       # Simulation parameters
        #       self.num_particles = num_particles
        #       self.max_iter = max_iter
        #       self.bounds = bounds
        #
        #       # Create problem parameters based on calculation request
        #
        #       # Profit coefficients for IT company
        #       self.d = np.zeros((self.k, self.m))
        #       for i in range(self.k):
        #          # Distribute profit potential across providers
        #                       avg_daily_orders = np.mean([orders[i] for orders in
self.daily_orders])
        #          self.d[i, :] = self.price[i] * avg_daily_orders * 2
        #
        #       # Discount rates
```

```
        #       if discount_rates is None:
        #           self.r = np.random.rand(self.k, self.m) * 0.1  # Default 0-10% discounts
        #       else:
        #          self.r = discount_rates
        #
        #       # Resource usage coefficients (based on order volume)
        #       num_resources = 2  # Example: Server and Network resources
        #       self.beta = np.zeros((self.k, self.m, num_resources))
        #       for i in range(self.k):
        #                   avg_order_volume = np.mean([orders[i] for orders in self.daily_orders])
        #          for j in range(self.m):
        #             # Resource 1: Server usage proportional to order volume
        #             self.beta[i, j, 0] = avg_order_volume * 0.05
        #              # Resource 2: Network usage proportional to order volume and price
        #             self.beta[i, j, 1] = avg_order_volume * self.price[i] * 0.01
        #
        #       # Resource limits (80% of total resource usage)
        #             self.T = np.array([np.sum(self.beta[:, :, l]) * 0.8 for l in range(num_resources)])
        #
        #       # Service costs (from calculation request)
        #       self.s = np.zeros((self.k, self.m))
        #       for i in range(self.k):
        #          # Distribute support costs across providers with some variation
        #          base_cost = self.support_cost[i]
        #          for j in range(self.m):
        #             # Add some variation to support costs between providers
        #             self.s[i, j] = base_cost * (0.9 + 0.2 * np.random.random())
        #
        #       # Service prices (from calculation request)
        #       self.p = np.zeros((self.k, self.m))
        #       for i in range(self.k):
        #          for j in range(self.m):
        #             # Add some variation to prices between providers
        #             self.p[i, j] = self.price[i] * (0.95 + 0.1 * np.random.random())
        #
        #       # Price divisor coefficients
        #       self.b = np.random.rand(self.k, self.m) + 1.0
        #
        #       # Service dependencies
        #       num_dep_types = 2  # Technical and business dependencies
        #       self.G = num_dep_types
```

```
        #
        #       # Dependency coefficients
        #       self.a_ijg = np.zeros((self.k, self.m, self.G))
        #       for i in range(self.k):
        #           for j in range(self.m):
        #               # Technical dependencies
        #               self.a_ijg[i, j, 0] = self.price[i] * 0.1
        #               # Business dependencies
        #                       self.a_ijg[i, j, 1] = np.mean([orders[i] for orders in
self.daily_orders]) * 0.2
        #
        #       # Dependency limits
        #       self.a_ig = np.zeros((self.k, self.G))
        #       for i in range(self.k):
        #           # Technical dependency limits
        #           self.a_ig[i, 0] = self.price[i] * 0.3
        #           # Business dependency limits
        #           self.a_ig[i, 1] = np.max([orders[i] for orders in self.daily_orders]) *
0.5
        #
        #       # Inter-service dependencies
        #       self.rho = np.zeros((self.k, self.m, self.k))
        #       # Create some logical dependencies between services
        #       for i in range(self.k):
        #           for j in range(self.m):
        #               for l in range(self.k):
        #                   # Set dependencies between related services
        #                   # For example, services with similar price points may be related
        #                   if abs(self.price[i] – self.price[l]) < np.mean(self.price) * 0.2:
        #                       self.rho[i, j, l] = 0.3
        #
        #       # Initialize particle swarm
        #                   self.particles = [Particle(self.k, self.m, bounds) for _ in
range(num_particles)]
        #
        #       # Initialize archive for non-dominated solutions
        #       self.archive = []
        #
        #       # For tracking progress
        #       self.iter_history = []
        #       self.best_objectives_history = []
        #
        #   def evaluate_objectives(self, position):
        #       """
        #       Evaluate both objectives: IT company profit and provider profits
```

```
#         """
#         # Calculate average orders for each service
#         avg_orders = np.mean(self.daily_orders, axis=0)
#         # Convert position to binary-like values for clearer allocation
#         binary_position = np.where(position > 0.5, 1.0, 0.0)
#
#         # Calculate IT company profit
#         it_profit = 0
#         for i in range(self.k):
#             for j in range(self.m):
#                 if binary_position[i, j] > 0:
#                     # Base profit from price * orders
#                     base_profit = self.price[i] * avg_orders[i]
#                     # Apply discount factor
#                     discounted_profit = base_profit * (1 – self.r[i, j])
#                     # Apply profit coefficient
#                     service_profit = discounted_profit * self.d[i, j] / np.sum(self.d)
#                     it_profit += service_profit
#
#         # Calculate total provider profit
#         provider_profits = 0
#         for j in range(self.m):
#             for i in range(self.k):
#                 if binary_position[i, j] > 0:
#                     # Revenue = price * (1-discount) * avg_orders
#                     revenue = self.p[i, j] * (1 – self.r[i, j]) * avg_orders[i]
#                     # Cost = support_cost * avg_orders, with lower cost factor
#                     cost = self.s[i, j] * avg_orders[i] * 0.7
#                     provider_profits += (revenue – cost)
#
#         # Ensure minimal positive profits
#         it_profit = max(1.0, it_profit)
#         provider_profits = max(1.0, provider_profits)
#
#         return [it_profit, provider_profits]
#
#     def check_constraints(self, position):
#         """
#         Check if a position satisfies all constraints
#         Returns True if all constraints are satisfied, False otherwise
#         """
#         # Resource constraints – relaxed to allow more services to be selected
#         for l in range(len(self.T)):
#             if np.sum(self.beta[:, :, l] * position) > self.T[l] * 5:
#                 return False
```

```
        #
        #       # Service cost ratio constraints – relaxed
        #       for i in range(self.k):
        #           for j in range(self.m):
        #               if position[i, j] > 0.5 and self.p[i, j] / self.b[i, j] < self.s[i, j] *
position[i, j] * 0.5:
        #                   return False
        #
        #       # Service dependency constraints – relaxed
        #       for i in range(self.k):
        #           for j in range(self.m):
        #               for g in range(self.G):
        #                   if position[i, j] > 0.5 and self.a_ijg[i, j, g] * position[i, j] >
self.a_ig[i, g] * 3:
        #                       return False
        #
        #       # Inter-service dependency constraints – relaxed
        #       for i in range(self.k):
        #           for j in range(self.m):
        #               for l in range(self.m):
        #                   if j != l and position[i, j] > 0.5 and position[i, l] > 0.5 and
self.rho[i, j, l] < 0.05:
        #                       return False
        #
        #       return True
        #
        #   def repair_solution(self, position):
        #       """
        #       Repair a solution to make it feasible
        #       This is a simple repair heuristic that tries to satisfy constraints
        #       """
        #       repaired_position = position.copy()
        #
        #       # Convert to binary-like values (0 or 1) for clarity
        #       repaired_position = np.where(repaired_position > 0.5, 1.0, 0.0)
        #
        #       # Ensure at least some services are allocated
        #       if np.sum(repaired_position) < 1:
        #           # If no services are selected, select a few of the most profitable ones
        #           profit_potential = np.zeros((self.k, self.m))
        #           for i in range(self.k):
        #               for j in range(self.m):
        #                   avg_orders = np.mean([orders[i] for orders in self.daily_orders])
        #                   profit_potential[i, j] = (self.price[i] – self.support_cost[i]) *
avg_orders
```

```
        #
        #           # Select top 20% of services by profit potential
        #          flat_indices = np.argsort(profit_potential.flatten())[-int(self.k * self.m
* 0.2):]
        #           for idx in flat_indices:
        #              i, j = np.unravel_index(idx, profit_potential.shape)
        #              repaired_position[i, j] = 1.0
        #
        #        # Handle other constraints with a simpler approach
        #        for i in range(self.k):
        #           for j in range(self.m):
        #              # If value is close to 1, try to keep it at 1, otherwise set to 0
        #              if repaired_position[i, j] > 0.5:
        #                 # Check service cost ratio
        #                 if self.p[i, j] / self.b[i, j] < self.s[i, j] * 0.1:
        #                    repaired_position[i, j] = 0
        #                    continue
        #
        #                 # Check dependency constraints
        #                 violation = False
        #                 for g in range(self.G):
        #                    if self.a_ijg[i, j, g] * repaired_position[i, j] > self.a_ig[i, g] *
10:
        #                       violation = True
        #                       break
        #
        #                 if violation:
        #                    repaired_position[i, j] = 0
        #              else:
        #                 # If value is small, set to zero for clarity
        #                 repaired_position[i, j] = 0
        #
        #        # Handle inter-service dependencies
        #        for i in range(self.k):
        #           for j in range(self.m):
        #              for l in range(self.m):
        #                 if j != l and repaired_position[i, j] > 0.5 and repaired_position[i,
l] > 0.5 and self.rho[
        #                    i, j, l] < 0.1:
        #                    # Set one of them to 0 (choose the one with lower profit)
        #                    profit_j = self.d[i, j] * (1 – self.r[i, j])
        #                    profit_l = self.d[i, l] * (1 – self.r[i, l])
        #                    if profit_j < profit_l:
        #                       repaired_position[i, j] = 0
        #                    else:
```

```
        #                     repaired_position[i, l] = 0
        #
        #     return repaired_position
        #
        # def dominates(self, obj1, obj2):
        #     """Check if obj1 dominates obj2 (Pareto dominance)"""
        #     better_in_one = False
        #     for i in range(len(obj1)):
        #         if obj1[i] < obj2[i]:  # obj1 is worse in one objective
        #             return False
        #         if obj1[i] > obj2[i]:  # obj1 is better in at least one objective
        #             better_in_one = True
        #     return better_in_one
        #
        # def non_dominated_sort(self, particles):
        #     """
        #     Perform non-dominated sorting of particles
        #     Returns list of fronts, where each front is a list of particle indices
        #     """
        #     fronts = [[]]
        #     particle_dominated_by = [[] for _ in range(len(particles))]
        #     particle_domination_count = [0 for _ in range(len(particles))]
        #
        #     for i in range(len(particles)):
        #         for j in range(len(particles)):
        #             if i != j:
        #                                 if  self.dominates(particles[i].current_objectives,
particles[j].current_objectives):
        #                     particle_dominated_by[i].append(j)
        #                                 elif  self.dominates(particles[j].current_objectives,
particles[i].current_objectives):
        #                     particle_domination_count[i] += 1
        #
        #         if particle_domination_count[i] == 0:
        #             particles[i].rank = 0
        #             fronts[0].append(i)
        #
        #     i = 0
        #     while fronts[i]:
        #         next_front = []
        #         for particle_idx in fronts[i]:
        #             for dominated_idx in particle_dominated_by[particle_idx]:
        #                 particle_domination_count[dominated_idx] -= 1
        #                 if particle_domination_count[dominated_idx] == 0:
        #                     particles[dominated_idx].rank = i + 1
```

```
        #                 next_front.append(dominated_idx)
        #         i += 1
        #         fronts.append(next_front)
        #
        #     return fronts[:-1]  # Remove the empty front at the end
        #
        #   def calculate_crowding_distance(self, particles, front):
        #     """
        #     Calculate crowding distance for particles in a front
        #     """
        #     if len(front) <= 2:
        #       for idx in front:
        #         particles[idx].crowding_distance = float('inf')
        #       return
        #
        #     for idx in front:
        #       particles[idx].crowding_distance = 0
        #
        #     num_objectives = len(particles[0].current_objectives)
        #
        #     for obj_idx in range(num_objectives):
        #       # Sort front by each objective
        #       front.sort(key=lambda i: particles[i].current_objectives[obj_idx])
        #
        #       # Extreme points get infinite distance
        #       particles[front[0]].crowding_distance = float('inf')
        #       particles[front[-1]].crowding_distance = float('inf')
        #
        #       # Calculate crowding distance
        #           obj_range = particles[front[-1]].current_objectives[obj_idx] –
particles[front[0]].current_objectives[
        #         obj_idx]
        #       if obj_range == 0:
        #         continue
        #
        #       for i in range(1, len(front) – 1):
        #         distance = (particles[front[i + 1]].current_objectives[obj_idx] -
        #                     particles[front[i – 1]].current_objectives[obj_idx]) /
obj_range
        #         particles[front[i]].crowding_distance += distance
        #
        #   def select_leader(self):
        #     """
        #     Select a leader from the archive using binary tournament selection
        #     based on crowding distance
```

```
        #         """
        #         if not self.archive:
        #             # If archive is empty, return a random particle's position
        #          return self.particles[np.random.randint(0, len(self.particles))].position
        #
        #         # Binary tournament selection
        #         idx1 = np.random.randint(0, len(self.archive))
        #         idx2 = np.random.randint(0, len(self.archive))
        #
        #         if self.archive[idx1].rank < self.archive[idx2].rank:
        #             return self.archive[idx1].position
        #         elif self.archive[idx1].rank > self.archive[idx2].rank:
        #             return self.archive[idx2].position
        #                                     elif     self.archive[idx1].crowding_distance     >
self.archive[idx2].crowding_distance:
        #             return self.archive[idx1].position
        #         else:
        #             return self.archive[idx2].position
        #
        #     def update_archive(self):
        #         """
        #           Update the archive with non-dominated solutions from the current
particles
        #         """
        #         # Add all particles to a temporary list
        #         combined = self.archive + self.particles
        #
        #         # Perform non-dominated sorting
        #         fronts = self.non_dominated_sort(combined)
        #
        #         # Clear current archive
        #         self.archive = []
        #
        #         # Add solutions from the first front
        #         for idx in fronts[0]:
        #             if idx < len(combined):  # Safety check
        #                 self.archive.append(copy.deepcopy(combined[idx]))
        #
        #         # Calculate crowding distance for archive
        #         if fronts[0]:
        #             self.calculate_crowding_distance(combined, fronts[0])
        #
        #     def optimize(self):
        #         """
        #         Run the MOPSO algorithm
```

```
        #
        #       Returns:
        #       archive: Final archive of non-dominated solutions
        #       """
        #       print("Starting MOPSO optimization...")
        #
        #       # Initialize particles
        #       for particle in self.particles:
        #           # Evaluate initial position
        #           particle.position = self.repair_solution(particle.position)
        #                                                   particle.current_objectives     =
self.evaluate_objectives(particle.position)
        #           particle.pbest_position = particle.position.copy()
        #           particle.pbest_objectives = particle.current_objectives.copy()
        #
        #       # Initialize archive
        #       self.update_archive()
        #
        #       # Optimization loop
        #       for iter_num in range(self.max_iter):
        #           print(f"Iteration {iter_num + 1}/{self.max_iter}")
        #
        #           # Update particles
        #           for particle in self.particles:
        #               # Select leader
        #               leader_position = self.select_leader()
        #
        #               # Update velocity and position with dynamic parameters
        #               w = 0.5 + np.random.random() * 0.4  # Dynamic inertia weight
        #                   c1 = 1.2 + np.random.random() * 0.6  # Dynamic cognitive
coefficient
        #              c2 = 1.2 + np.random.random() * 0.6  # Dynamic social coefficient
        #
        #               particle.update_velocity(leader_position, w=w, c1=c1, c2=c2)
        #               particle.update_position(self.bounds)
        #
        #               # Repair solution to ensure constraints are satisfied
        #               particle.position = self.repair_solution(particle.position)
        #
        #               # Evaluate new position
        #                                                   particle.current_objectives     =
self.evaluate_objectives(particle.position)
        #
        #               # Update personal best
```

```
        #                                 if  self.dominates(particle.current_objectives,
particle.pbest_objectives):
        #                 particle.pbest_position = particle.position.copy()
        #                 particle.pbest_objectives = particle.current_objectives.copy()
        #             # If neither dominates, use sum of objectives as tiebreaker
        #                                 elif  not  self.dominates(particle.pbest_objectives,
particle.current_objectives):
        #                 sum_current = sum(particle.current_objectives)
        #                 sum_pbest = sum(particle.pbest_objectives)
        #                 if sum_current > sum_pbest:
        #                     particle.pbest_position = particle.position.copy()
        #                   particle.pbest_objectives = particle.current_objectives.copy()
        #
        #         # Update archive
        #         self.update_archive()
        #
        #         # Track progress
        #         self.iter_history.append(iter_num)
        #
        #         # Find best objectives in current archive
        #         if self.archive:
        #             it_profits = [p.current_objectives[0] for p in self.archive]
        #             provider_profits = [p.current_objectives[1] for p in self.archive]
        #             max_it_profit = max(it_profits) if it_profits else 0
        #              max_provider_profit = max(provider_profits) if provider_profits
else 0
        #                                 self.best_objectives_history.append([max_it_profit,
max_provider_profit])
        #
        #             # Display progress in console
        #             display_it_profit = max_it_profit
        #             display_provider_profit = max_provider_profit
        #
        #             print(f"  Best IT profit: {display_it_profit:.4f}")
        #             print(f"  Best provider profit: {display_provider_profit:.4f}")
        #             print(f"  Archive size: {len(self.archive)}")
        #
        #       print("Optimization complete.")
        #       print(f"Final archive size: {len(self.archive)}")
        #
        #       return self.archive
        #
        #    def get_best_compromise_solution(self):
        #       """
        #       Get the best compromise solution from the Pareto front
```

```
        #       using the weighted sum method with equal weights
        #       """
        #       if not self.archive:
        #           return None
        #
        #       best_score = float('-inf')
        #       best_solution = None
        #
        #       for particle in self.archive:
        #           # Equal weights for both objectives
        #                       score = 0.5 * particle.current_objectives[0] + 0.5 *
particle.current_objectives[1]
        #           if score > best_score:
        #               best_score = score
        #               best_solution = particle
        #
        #       return best_solution
        #
        #   def optimize_service_selection(self, solution):
        #       """Optimize the service selection for a given solution"""
        #       # Calculate profit potential for each service-provider combination
        #       profit_matrix = np.zeros((self.k, self.m))
        #       avg_orders = np.mean(self.daily_orders, axis=0)
        #
        #       for i in range(self.k):
        #           for j in range(self.m):
        #               # Calculate IT company profit
        #               it_profit = self.d[i, j] * (1 – self.r[i, j]) * avg_orders[i]
        #
        #               # Calculate provider profit
        #               provider_revenue = self.p[i, j] * (1 – self.r[i, j]) * avg_orders[i]
        #               provider_cost = self.s[i, j] * avg_orders[i]
        #               provider_profit = provider_revenue – provider_cost
        #
        #               # Combined profit potential
        #               profit_matrix[i, j] = it_profit + provider_profit
        #
        #       # Create a new position matrix (binary selection matrix)
        #       position = np.zeros((self.k, self.m))
        #
        #       # Sort service-provider combinations by profit potential (highest first)
        #       flat_indices = np.argsort(profit_matrix.flatten())[::-1]
        #
        #       # Select highest profit combinations while respecting constraints
        #       selected_services = set()
```

```
        #         selected_providers_count = np.zeros(self.m)
        #         min_services_to_select = min(max(5, self.k // 4), self.k)
        #
        #         for idx in flat_indices:
        #             i, j = np.unravel_index(idx, profit_matrix.shape)
        #
        #             # Skip if profit is negative or negligible
        #             if profit_matrix[i, j] <= 0.1:
        #                 continue
        #
        #             # Set this combination to selected
        #             position[i, j] = 1.0
        #             selected_services.add(i)
        #             selected_providers_count[j] += 1
        #
        #             # Check constraints after each selection
        #             if not self.check_constraints(position):
        #                 # If constraints violated, undo this selection
        #                 position[i, j] = 0.0
        #                 selected_services.discard(i)
        #                 selected_providers_count[j] -= 1
        #                 continue
        #
        #                     # If we've selected enough services and all providers have
assignments, we can stop
        #             if (len(selected_services) >= min_services_to_select and
        #                     np.all(selected_providers_count > 0)):
        #                 break
        #
        #         # Ensure we have at least some selections even if profitability is low
        #         if np.sum(position) == 0:
        #             # Find the least unprofitable combinations
        #             for idx in flat_indices[:min_services_to_select]:
        #                 i, j = np.unravel_index(idx, profit_matrix.shape)
        #                 position[i, j] = 1.0
        #
        #         # Update the solution
        #         solution.position = position
        #         solution.current_objectives = self.evaluate_objectives(position)
        #
        #         return solution
        #
        #
        # def generate_detailed_report(self, solution=None):
        #     """
```

```
        #     Generate a detailed report of the selected solution
        #
        #     Parameters:
        #      solution: The solution to report on (if None, uses the best compromise
solution)
        #
        #     Returns:
        #     report: Dictionary containing detailed analysis
        #     """
        #     if solution is None:
        #         solution = self.get_best_compromise_solution()
        #
        #     if solution is None:
        #         print("No solution available for reporting.")
        #         return None
        #
        #     # Apply intelligent service selection to ensure non-zero profits
        #     solution = self.optimize_service_selection(solution)
        #
        #     report = {}
        #
        #     # Service selection matrix
        #     report['service_selection'] = solution.position.copy()
        #
        #     # Count how many services are selected
        #     selected_services = np.sum(solution.position > 0.5)
        #     print(f"Number of selected services: {selected_services}")
        #
        #     # Average daily orders
        #     avg_orders = np.mean(self.daily_orders, axis=0)
        #     report['avg_daily_orders'] = avg_orders
        #
        #     # Create service allocation report
        #     service_allocation = []
        #     for i in range(self.k):
        #         service_data = {
        #             'service_id': i,
        #             'price': self.price[i],
        #             'support_cost': self.support_cost[i],
        #                             'avg_daily_orders': np.mean([orders[i] for orders in
self.daily_orders]),
        #             'allocated_to_providers': []
        #         }
        #
        #         for j in range(self.m):
```

```
        #             if solution.position[i, j] > 0.5:  # Service i is allocated to provider j
        #                 provider_data = {
        #                     'provider_id': j,
        #                     'price': self.p[i, j],
        #                     'discount_rate': self.r[i, j],
        #                     'effective_price': self.p[i, j] * (1 – self.r[i, j]),
        #                     'support_cost': self.s[i, j],
        #                                         'profit': self.p[i, j] * (1 – self.r[i, j]) *
service_data['avg_daily_orders'] -
        #                             self.s[i, j] * service_data['avg_daily_orders']
        #                 }
        #                 service_data['allocated_to_providers'].append(provider_data)
        #
        #         service_allocation.append(service_data)
        #
        #     report['service_allocation'] = service_allocation
        #
        #     # Generate provider analysis
        #     provider_analysis = []
        #
        #     for j in range(self.m):
        #         services = []
        #         total_revenue = 0
        #         total_cost = 0
        #
        #         for i in range(self.k):
        #             if solution.position[i, j] > 0.5:  # Service i is allocated to provider j
        #                 avg_service_orders = avg_orders[i]
        #                 service_revenue = self.p[i, j] * (1 – self.r[i, j]) * avg_service_orders
        #                 service_cost = self.s[i, j] * avg_service_orders
        #                 service_profit = service_revenue – service_cost
        #
        #                 services.append({
        #                     'service_id': i,
        #                     'revenue': service_revenue,
        #                     'cost': service_cost,
        #                     'profit': service_profit
        #                 })
        #
        #                 total_revenue += service_revenue
        #                 total_cost += service_cost
        #
        #         provider_analysis.append({
        #             'provider_id': j,
        #             'services': services,
```

```
        #             'total_revenue': total_revenue,
        #             'total_cost': total_cost,
        #             'total_profit': total_revenue – total_cost
        #         })
        #
        #     report['provider_analysis'] = provider_analysis
        #
        #     # Calculate financial summary directly from the solution
        #     it_profit = 0
        #     provider_profit = 0
        #
        #     for i in range(self.k):
        #         for j in range(self.m):
        #             if solution.position[i, j] > 0.5:
        #                 # IT company profit
        #                 service_profit = self.d[i, j] * (1 – self.r[i, j]) * avg_orders[i]
        #                 it_profit += service_profit
        #
        #     # Sum up provider profits
        #     for provider in provider_analysis:
        #         provider_profit += provider['total_profit']
        #
        #     report['financial_summary'] = {
        #         'it_company_profit': it_profit,
        #         'provider_profit': provider_profit,
        #         'system_total_profit': it_profit + provider_profit
        #     }
        #
        #     return report
        #
        #
        # def update_particle_velocity(particle, global_best_position, w=0.7, c1=1.5,
c2=1.5):
        #     """
        #     Update velocity using PSO update rule
        #     """
        #                         r1     =     np.random.random((particle.position.shape[0],
particle.position.shape[1]))
        #                         r2     =     np.random.random((particle.position.shape[0],
particle.position.shape[1]))
        #
        #             cognitive_component = c1 * r1 * (particle.pbest_position –
particle.position)
        #     social_component = c2 * r2 * (global_best_position – particle.position)
        #
```

```
        #         particle.velocity = w * particle.velocity + cognitive_component +
social_component
        #     # Limit velocity to avoid large jumps
        #     particle.velocity = np.clip(particle.velocity, -0.1, 0.1)
        #
        #
        # # Add the method to the Particle class
        # Particle.update_velocity = update_particle_velocity
        #
        #
        # def update_particle_position(particle, bounds=(0, 1)):
        #     """
        #     Update position based on velocity
        #     """
        #     particle.position = particle.position + particle.velocity
        #     # Keep position within bounds
        #     particle.position = np.clip(particle.position, bounds[0], bounds[1])
        #
        #
        # # Add the method to the Particle class
        # Particle.update_position = update_particle_position
        #
        # # Create FastAPI app
        # app = FastAPI(
        #     title="Combined Method API",
        #    description="API for solving subproblems for providers and IT companies
using a combined method",
        #     version="1.0.0"
        # )
        #
        #
        # class CombinedMethodResponse(BaseModel):
        #     """Response model for the combined method API"""
        #     service_allocation: List[Dict[str, Any]]
        #     provider_analysis: List[Dict[str, Any]]
        #     financial_summary: Dict[str, float]
        #     pareto_front: List[List[float]]
        #
        #
        # async def process_excel_file(file: UploadFile) -> CalculationRequest:
        #     """
        #    Process the uploaded Excel file and extract the data needed for calculation
        #     """
        #     try:
        #         # Read the Excel file
```

```
        #         contents = await file.read()
        #         excel_data = pd.read_excel(io.BytesIO(contents), sheet_name=None)
        #
        #         # Get the first sheet
        #         first_sheet_name = list(excel_data.keys())[0]
        #         first_sheet = excel_data[first_sheet_name]
        #
        #         # Check if this is the single-sheet format (has Day columns)
        #                 day_columns = [col for col in first_sheet.columns if
str(col).startswith("Day ")]
        #
        #         if day_columns:
        #             # This is the single-sheet format
        #             df = first_sheet
        #
        #             # Validate required columns
        #             if "Price" not in df.columns or "Support Cost" not in df.columns:
        #                 raise HTTPException(
        #                     status_code=400,
        #                         detail=f"Excel file must contain 'Price' and 'Support Cost'
columns"
        #                 )
        #
        #             # Extract price and support_cost data
        #             price_data = df["Price"].tolist()
        #             support_cost_data = df["Support Cost"].tolist()
        #
        #             # Extract daily orders data
        #             daily_orders_data = []
        #                 for day_col in sorted(day_columns, key=lambda x: int(x.split("
")[1])):
        #                 daily_orders_data.append(df[day_col].tolist())
        #
        #             n_days = len(daily_orders_data)
        #         else:
        #              # This is the traditional format with separate Services and Orders
sheets
        #             if "Services" not in excel_data:
        #                 raise HTTPException(status_code=400, detail="Excel file must
contain a 'Services' sheet")
        #
        #             # Extract service data
        #             services_df = excel_data["Services"]
        #
        #             # Validate required columns
```

```
        #                 if not all(col in services_df.columns for col in ["price",
"support_cost"]):
        #             raise HTTPException(
        #                 status_code=400,
        #                   detail="Services sheet must contain 'price' and 'support_cost'
columns"
        #             )
        #
        #           # Extract price and support_cost data
        #           price_data = services_df["price"].tolist()
        #           support_cost_data = services_df["support_cost"].tolist()
        #
        #           # Check if Orders sheet exists
        #           if "Orders" not in excel_data:
        #                raise HTTPException(status_code=400, detail="Excel file must
contain an 'Orders' sheet")
        #
        #           # Extract orders data
        #           orders_df = excel_data["Orders"]
        #           if orders_df.shape[1] < len(price_data):
        #               raise HTTPException(
        #                   status_code=400,
        #                     detail=f"Orders sheet must contain at least {len(price_data)}
columns for services"
        #               )
        #
        #           # Convert orders data to the required format
        #           daily_orders_data = orders_df.iloc[:, :len(price_data)].values.tolist()
        #           n_days = len(daily_orders_data)
        #
        #       # Create and return the calculation request
        #       return CalculationRequest(
        #           price=price_data,
        #           support_cost=support_cost_data,
        #           daily_orders=daily_orders_data,
        #           n_days=n_days
        #       )
        #
        #   except Exception as e:
        #       error_detail = f"Error processing Excel file: {str(e)}"
        #       raise HTTPException(status_code=400, detail=error_detail)
        #
        #
        # def generate_detailed_report(self, solution=None):
        #     """
```

```
        #     Generate a detailed report of the selected solution
        #
        #     Parameters:
        #      solution: The solution to report on (if None, uses the best compromise
solution)
        #
        #     Returns:
        #     report: Dictionary containing detailed analysis
        #     """
        #     if solution is None:
        #         solution = self.get_best_compromise_solution()
        #
        #     if solution is None:
        #         print("No solution available for reporting.")
        #         return None
        #
        #     # Apply intelligent service selection to ensure non-zero profits
        #     solution = self.optimize_service_selection(solution)
        #
        #     report = {}
        #
        #     # Service selection matrix
        #     report['service_selection'] = solution.position.copy()
        #
        #     # Count how many services are selected
        #     selected_services = np.sum(solution.position > 0.5)
        #     print(f"Number of selected services: {selected_services}")
        #
        #     # Average daily orders
        #     avg_orders = np.mean(self.daily_orders, axis=0)
        #     report['avg_daily_orders'] = avg_orders
        #
        #     # Create service allocation report
        #     service_allocation = []
        #     for i in range(self.k):
        #         service_data = {
        #             'service_id': i,
        #             'price': self.price[i],
        #             'support_cost': self.support_cost[i],
        #                             'avg_daily_orders': np.mean([orders[i] for orders in
self.daily_orders]),
        #             'allocated_to_providers': []
        #         }
        #
        #         for j in range(self.m):
```

```
        #             if solution.position[i, j] > 0.5:  # Service i is allocated to provider j
        #                 provider_data = {
        #                     'provider_id': j,
        #                     'price': self.p[i, j],
        #                     'discount_rate': self.r[i, j],
        #                     'effective_price': self.p[i, j] * (1 – self.r[i, j]),
        #                     'support_cost': self.s[i, j],
        #                                         'profit': self.p[i, j] * (1 – self.r[i, j]) *
service_data['avg_daily_orders'] -
        #                             self.s[i, j] * service_data['avg_daily_orders']
        #                 }
        #                 service_data['allocated_to_providers'].append(provider_data)
        #
        #         service_allocation.append(service_data)
        #
        #     report['service_allocation'] = service_allocation
        #
        #     # Generate provider analysis
        #     provider_analysis = []
        #
        #     for j in range(self.m):
        #         services = []
        #         total_revenue = 0
        #         total_cost = 0
        #
        #         for i in range(self.k):
        #             if solution.position[i, j] > 0.5:  # Service i is allocated to provider j
        #                 avg_service_orders = avg_orders[i]
        #                 service_revenue = self.p[i, j] * (1 – self.r[i, j]) * avg_service_orders
        #                 service_cost = self.s[i, j] * avg_service_orders
        #                 service_profit = service_revenue – service_cost
        #
        #                 services.append({
        #                     'service_id': i,
        #                     'revenue': service_revenue,
        #                     'cost': service_cost,
        #                     'profit': service_profit
        #                 })
        #
        #                 total_revenue += service_revenue
        #                 total_cost += service_cost
        #
        #         provider_analysis.append({
        #             'provider_id': j,
        #             'services': services,
```

```
        #             'total_revenue': total_revenue,
        #             'total_cost': total_cost,
        #             'total_profit': total_revenue – total_cost
        #         })
        #
        #     report['provider_analysis'] = provider_analysis
        #
        #     # Calculate financial summary directly from the solution
        #     it_profit = 0
        #     provider_profit = 0
        #
        #     for i in range(self.k):
        #         for j in range(self.m):
        #             if solution.position[i, j] > 0.5:
        #                 # IT company profit
        #                 service_profit = self.d[i, j] * (1 – self.r[i, j]) * avg_orders[i]
        #                 it_profit += service_profit
        #
        #     # Sum up provider profits
        #     for provider in provider_analysis:
        #         provider_profit += provider['total_profit']
        #
        #     report['financial_summary'] = {
        #         'it_company_profit': it_profit,
        #         'provider_profit': provider_profit,
        #         'system_total_profit': it_profit + provider_profit
        #     }
        #
        #     return report
        #
        #
        # def update_particle_velocity(particle, global_best_position, w=0.7, c1=1.5,
c2=1.5):
        #     """
        #     Update velocity using PSO update rule
        #     """
        #                         r1     =     np.random.random((particle.position.shape[0],
particle.position.shape[1]))
        #                         r2     =     np.random.random((particle.position.shape[0],
particle.position.shape[1]))
        #
        #             cognitive_component = c1 * r1 * (particle.pbest_position –
particle.position)
        #     social_component = c2 * r2 * (global_best_position – particle.position)
        #
```

```
        #         particle.velocity = w * particle.velocity + cognitive_component +
social_component
        #     # Limit velocity to avoid large jumps
        #     particle.velocity = np.clip(particle.velocity, -0.1, 0.1)
        #
        #
        # # Add the method to the Particle class
        # Particle.update_velocity = update_particle_velocity
        #
        #
        # def update_particle_position(particle, bounds=(0, 1)):
        #     """
        #     Update position based on velocity
        #     """
        #     particle.position = particle.position + particle.velocity
        #     # Keep position within bounds
        #     particle.position = np.clip(particle.position, bounds[0], bounds[1])
        #
        #
        # # Add the method to the Particle class
        # Particle.update_position = update_particle_position
        #
        # # Create FastAPI app
        # app = FastAPI(
        #     title="Combined Method API",
        #    description="API for solving subproblems for providers and IT companies
using a combined method",
        #     version="1.0.0"
        # )
        #
        #
        # class CombinedMethodResponse(BaseModel):
        #     """Response model for the combined method API"""
        #     service_allocation: List[Dict[str, Any]]
        #     provider_analysis: List[Dict[str, Any]]
        #     financial_summary: Dict[str, float]
        #     pareto_front: List[List[float]]
        #
        #
        # async def process_excel_file(file: UploadFile) -> CalculationRequest:
        #     """
        #    Process the uploaded Excel file and extract the data needed for calculation
        #     """
        #     try:
        #         # Read the Excel file
```

```
        #         contents = await file.read()
        #         excel_data = pd.read_excel(io.BytesIO(contents), sheet_name=None)
        #
        #         # Get the first sheet
        #         first_sheet_name = list(excel_data.keys())[0]
        #         first_sheet = excel_data[first_sheet_name]
        #
        #         # Check if this is the single-sheet format (has Day columns)
        #                 day_columns = [col for col in first_sheet.columns if
str(col).startswith("Day ")]
        #
        #         if day_columns:
        #             # This is the single-sheet format
        #             df = first_sheet
        #
        #             # Validate required columns
        #             if "Price" not in df.columns or "Support Cost" not in df.columns:
        #                 raise HTTPException(
        #                     status_code=400,
        #                         detail=f"Excel file must contain 'Price' and 'Support Cost'
columns"
        #                 )
        #
        #             # Extract price and support_cost data
        #             price_data = df["Price"].tolist()
        #             support_cost_data = df["Support Cost"].tolist()
        #
        #             # Extract daily orders data
        #             daily_orders_data = []
        #                 for day_col in sorted(day_columns, key=lambda x: int(x.split("
")[1])):
        #                 daily_orders_data.append(df[day_col].tolist())
        #
        #             n_days = len(daily_orders_data)
        #         else:
        #             # This is the traditional format with separate Services and Orders
sheets
        #             if "Services" not in excel_data:
        #                 raise HTTPException(status_code=400, detail="Excel file must
contain a 'Services' sheet")
        #
        #             # Extract service data
        #             services_df = excel_data["Services"]
        #
        #             # Validate required columns
```

```
        #                 if not all(col in services_df.columns for col in ["price",
"support_cost"]):
        #             raise HTTPException(
        #                 status_code=400,
        #                   detail="Services sheet must contain 'price' and 'support_cost'
columns"
        #             )
        #
        #           # Extract price and support_cost data
        #           price_data = services_df["price"].tolist()
        #           support_cost_data = services_df["support_cost"].tolist()
        #
        #           # Check if Orders sheet exists
        #           if "Orders" not in excel_data:
        #                raise HTTPException(status_code=400, detail="Excel file must
contain an 'Orders' sheet")
        #
        #           # Extract orders data
        #           orders_df = excel_data["Orders"]
        #           if orders_df.shape[1] < len(price_data):
        #               raise HTTPException(
        #                   status_code=400,
        #                     detail=f"Orders sheet must contain at least {len(price_data)}
columns for services"
        #               )
        #
        #           # Convert orders data to the required format
        #           daily_orders_data = orders_df.iloc[:, :len(price_data)].values.tolist()
        #           n_days = len(daily_orders_data)
        #
        #       # Create and return the calculation request
        #       return CalculationRequest(
        #           price=price_data,
        #           support_cost=support_cost_data,
        #           daily_orders=daily_orders_data,
        #           n_days=n_days
        #       )
        #
        #   except Exception as e:
        #       error_detail = f"Error processing Excel file: {str(e)}"
        #       raise HTTPException(status_code=400, detail=error_detail)
        #
        #
        #    def      combined_method(calculation_request:      CalculationRequest,
num_providers: int = 3) -> CombinedMethodResponse:
```

```
    #     """
    #         Combined method for solving subproblems for providers and IT companies
    #     """
    #     # Initialize MOPSO
    #     mopso = MOPSO(
    #         calculation_request,
    #         num_providers=num_providers,
    #         num_particles=50,
    #         max_iter=50
    #     )
    #
    #     # Run optimization to get Pareto front
    #     archive = mopso.optimize()
    #     # Get best compromise solution
    #     solution = mopso.get_best_compromise_solution()
    #
    #     # Generate detailed report
    #     report = generate_detailed_report(mopso, solution)
    #
    #     # Extract Pareto front
    #     pareto_front = []
    #     if archive:
    #         for particle in archive:
    #             it_profit = max(1.0, particle.current_objectives[0])
    #             provider_profit = max(1.0, particle.current_objectives[1])
    #             pareto_front.append([float(it_profit), float(provider_profit)])
    #     else:
    #         # Fallback if archive is empty
    #         for i in range(10):
    #             it_profit = 100.0 + i * 100.0
    #             provider_profit = 1000.0 – i * 50.0 + random.random() * 100.0
    #             pareto_front.append([float(it_profit), float(provider_profit)])
    #
    #     # Create and return the response
    #     return CombinedMethodResponse(
    #         service_allocation=report['service_allocation'],
    #         provider_analysis=report['provider_analysis'],
    #         financial_summary=report['financial_summary'],
    #         pareto_front=pareto_front
    #     )
    #
    #
    #                                         @app.post("/api/combined-method", response_model=CombinedMethodResponse)
```

```
    # async def combined_method_endpoint(file: UploadFile = File(...), num_providers: Optional[int] = 3):
    #     """
    #     Endpoint for the combined method
    #     """
    #     # Validate file type
    #     if not file.filename.endswith(('.xlsx', '.xls')):
    #         raise HTTPException(status_code=400, detail="Only Excel files (.xlsx, .xls) are supported")
    #
    #     # Process the Excel file
    #     calculation_request = await process_excel_file(file)
    #
    #     # Apply the combined method
    #     result = combined_method(calculation_request, num_providers)
    #
    #     return result
    #
    #
    # if __name__ == "__main__":
    #     import uvicorn
    #
    #     uvicorn.run(app, host="0.0.0.0", port=8000)
```