НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова
праця на правах рукопису

ДУДНИК ОЛЕКСІЙ ОЛЕКСАНДРОВИЧ

УДК 338.24.01

ДИСЕРТАЦІЯ

ІННОВАЦІЙНІ ТЕХНОЛОГІЇ В УПРАВЛІННІ ПОРТФЕЛЕМ ПРОЄКТІВ ІТ-КОМПАНІЙ

Спеціальність 051– Економіка
Галузь знань 05 – Соціальні та поведінкові науки

Подається на здобуття наукового ступеня доктора філософії
Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ О.О. Дудник

Науковий керівник – Соколовська Зоя Миколаївна, доктор економічних наук, професор

Одеса 2023

АНОТАЦІЯ

Дудник О.О. Інноваційні технології в управлінні портфелем проєктів ІТ-компаній. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 051 – Економіка. – Національний університет “Одеська політехніка” МОН України, Одеса, 2023.

Дисертаційну роботу присвячено застосуванню інноваційних технологій в процесі управління аутсорсингових ІТ-компаній в якості інструментально-аналітичного забезпечення ефективного управління ІТ-проєктами та обґрунтування економічних рішень на основі їх експертної оцінки.

На сьогоднішній день в Україні і світі наявна велика кількість аутсорсингових ІТ-компаній, які відповідають за розробку досить об'ємного сектору ІТ-проєктів. Процесам управління проєктами у межах аутсорсингової ІТ-компанії притаманна низка особливостей, найважливішими з яких є підвищення відповідальності стосовно підпроцесу розробки програмного забезпечення, знижена або взагалі відсутня відповідальність за усі інші підпроцеси та часткова фінансова залежність від компанії замовника.

У зв'язку з цим зростає інтерес до інноваційних підходів у розробленні інструментально-аналітичного забезпечення ефективного управління ІТ-проєктами та обґрунтування економічних рішень на основі експертної оцінки поточного статусу цих проєктів, що в свою чергу підвищує ефективність підпроцесу розробки програмного забезпечення, дозволяючи мінімізувати негативні наслідки фінансової залежності від компанії замовника.

У розділі 1 розглянуто теоретичні засади управління ІТ-проєктами та їх портфелями. Демонструється важливість економічних показників та метрик успішності проєкту для досягнення успіху у проєктному менеджменті. Акцентується увага на важливості врахування не лише технічних, але й бізнес-аспектів ІТ-проєкту.

Проведено аналіз основних трендів інформаційної економіки. Встановлено, що світовими трендами є зростання обсягу даних, розвиток технологій штучного інтелекту та інтернету речей. Україна не є винятком і також активно розвивається в галузі ІТ. Роль інформаційної економіки в Україні є дуже важливою, оскільки ця галузь є джерелом збільшення ВВП та прискорення економічного розвитку. Напруження на глобальній арені, зокрема Російсько-Українська війна та пандемія COVID-19, також мають значний вплив на інформаційну економіку України. Це стало причиною серйозних викликів для бізнесу, зокрема для ІТ-компаній. Наприклад, падіння попиту на послуги та продукти, або недостатність фінансування можуть стати причиною кадрових скорочень та обмеження можливостей для виконання нових ІТ-проєктів. У світлі цих факторів набувають значення ефективні технології та процеси управління ІТ-проєктами, які дозволяють швидко реагувати на зміни і мінімізувати ризики.

Визначено основні проблеми та ризики управління проєктами в ІТ-компаніях. З одного боку, швидкий розвиток технологій може бути корисним для ІТ-компаній, адже нові інструменти та технології забезпечують більш ефективне виконання проєктів і зменшують витрати на програмну розробку. Однак, з іншого боку, постійні зміни створюють проблеми й ризики в управлінні проєктами: наприклад, зміни в ринкових умовах, технічні проблеми, недостатня кваліфікація персоналу, неправильне планування ресурсів та багато інших чинників. Окрім того, інформаційна економіка має вплив на ринкові умови та конкуренцію в галузі. Сучасні технології дозволяють ІТ-компаніям прискорити програмну розробку, що створює більш жорстку конкуренцію та може впливати на ціну ІТ-послуг, оскільки інноваційні рішення зазвичай коштують більше, ніж традиційні. Таким чином, у разі неефективного управління ІТ-проєктами може виникнути затримка у їх виконанні, збільшення витрат або навіть повний провал проєктів.

Досліджено основні технології управління ІТ-проєктами. Серед них виділено Agile технологію та Stage-Gate фреймворк. Agile технологія заснована

на ітераційному інкрементальному розробленні програмного продукту та активній участі замовника у процесі розробки. Перевагою використання цієї технології є швидке реагування на зміни вимог замовника та зменшення часу від розробки до випуску готового продукту. Однак, її недоліками є необхідність високого рівня комунікації та координації між членами команди розробників та замовником, що може призвести до затримок у виконанні ІТ-проєкту. Stage-Gate фреймворк, розроблений Робертом Купером, заснований на послідовному проходженні етапів розробки продукту, з встановленням точок прийняття рішень щодо переходу до наступних етапів. Його перевагами є більш структурований та прогнозований процес програмної розробки, що дозволяє більш точно визначити результати кожного етапу. Однак, недоліками є більш складна процедура ухвалення рішень на кожному етапі, що може призвести до затримок у виконанні проєкту.

Результати досліджень, відображені у розділі 1, дозволяють зробити висновок, що використання нечіткої логіки для розв’язання завдань управління ІТ-проєктами є достатньо рідким явищем. Більшу увагу приділяють процесу взагалі, будуючи різноманітні фреймворки, націлені на загальне описання процесу, а також на його подрібнення на менші завдання. Але процедури розв’язання цих завдань дослідники вивчають значно менше. Наявні лише окремі дослідження щодо підзавдань процесу з використанням нечіткої логіки; майже не має прецедентів використання нечіткої логіки стосовно процесу в цілому. Використання нечіткої логіки для розв’язання задач управління ІТ-проєктами може бути ефективним, оскільки нечітка логіка дозволяє враховувати не тільки точні значення параметрів, а й їх нечіткі значення, що може покращити ступінь обґрунтованості та загальні результати прийняття економічних рішень; спростити процес їх реалізації.

В умовах зростаючої конкуренції на ринку стає все більш складною задача управління ІТ-проєктами. Управління портфелем ІТ-проєктів вимагає розуміння досить великої кількості факторів, які впливають на ефективність проєктування та розробки програмного забезпечення. Інструментально-

аналітичне забезпечення є надзвичайно важливим інструментом управління IT-проєктами. Використання апарату нечіткої логіки дає змогу враховувати нечіткість даних та знань, що має важливе значення в умовах нестабільності процесу управління та значного ступеня невизначеності у взаємодії між учасниками проєкту.

У зв'язку з цим *у розділі 2* запропонована концептуальна модель обґрунтування й прийняття економічних рішень з управління проєктами в умовах невизначеного середовища і неточної інформації. Розглянуто основні теоретичні засади нечіткої логіки як математичного апарату, призначеного для роботи з нечіткими, неоднозначними, частковими та неповними даними. Розглянуто поняття нечіткої множини, функцій належності та нечіткого оператору. Виділено основні переваги нечітких інтелектуальних систем, реалізованих на основі математичного апарату нечіткої логіки.

Прикладна інструментально-аналітична база реалізації моделі обґрунтування й прийняття економічних рішень – нечітка експертна система FuzzyKIDE: авторська програмна оболонка, реалізована на основі нечіткої логіки. В основі архітектури FuzzyKIDE лежить фазифікація даних, що дозволяє представити кожен елемент в базі даних у вигляді функції належності до деякої множини з бази знань. У порівнянні з іншими нечіткими експертними системами FuzzyKIDE використовує оптимізовану модель бази знань. Знання системи відносно об’єкту функціонування, що досліджується, описуються за допомогою нечітких правил виводу, лінгвістичних змінних та функцій належності. Кожна функція описує, наскільки елемент відповідає певній характеристиці, тобто описує ступінь належності до деякої нечіткої множини. Нечіткі правила визначають зв'язки між вхідними та вихідними лінгвістичними змінними. Такий підхід дає змогу легко модифікувати базу знань, додавати нові правила або вносити зміни в існуючі. Ще однією особливістю є використання SNEPS семантичних мереж у ядрі нечіткого виводу, що дозволяє зменшити кількість звернень до бази знань і бази даних та підвищити ефективність використання знань системи. Таким чином, FuzzyKIDE має багато переваг

порівняно з класичними експертними системами, зокрема, структуровану модель бази знань та можливість використання семантичних мереж для оптимального зберігання та використання знань.

Запропоновано модифікацію Agile Stage-Gate фреймворку з використанням нечітких систем, що дозволяє врахувати особливості IT-проєктів в цілому та особливості сучасних ітеративних циклів розробки. Особливістю запропонованої схеми є використання систем нечіткої логіки при переході від брам до наступних етапів. У способі розстановки брам між етапами є одна суттєва особливість − деякі з брам не є однонаправленими. Наприклад, брама після процесу пошуку проєкту є однонаправленою, адже сам процес пошуку проєкту не є ітераційним. Водночас, брама перед процесом Agile спринту є двосторонньою, адже після кожного спринту проходить процес оцінки результатів на цій же брамі з метою визначити, чи є необхідність повторного переходу на попередній етап.

Запропонований у розділі математичний апарат спрямований на рішення широкого кола задач, пов’язаних з різноманітними аспектами обґрунтування стратегічних та оперативних економічних рішень, що потребують проведення комплексного аналізу впливу факторів різної природи; врахування ентропії та неповноти/неточності інформації стосовно внутрішнього й зовнішнього середовища досліджуваних процесів.

Працездатність розробленої інструментально-аналітичної бази прийняття рішень продемонстрована *в розділі 3* на результатах, отриманих в ході розв’язання типових задач управління IT-проєктами в аутсорсингових компаніях: пошук проєктів та формування складу й структури портфоліо; моніторинг поточного статусу проєктів; стратегічної орієнтації IT-компанії (професійного спрямування та максимізації прибутку). Проведений порівняльний аналіз наявних історичних даних та даних, отриманих в результаті експлуатації програмної оболонки FuzzyKIDE, реалізованої в контурі модифікованого Agile Stage-Gate фреймворку, довів ефективність запропонованого підходу. Порівняння показало, що використання програмної

оболонки значно підвищує ефективність процесу прийняття рішень і призводить до якісного покращення абсолютних та відносних економічних показників діяльності аутсорсингової ІТ-компанії.

Подальша експлуатація розробленого математичного апарату призведе до збагачення баз знань та даних, що сприятиме розвитку інформаційного забезпечення процесів підтримки прийняття обґрунтованих економічних рішень щодо управління ІТ-проєктами/ портфоліо проєктів.

Результати дисертаційної роботи було впроваджено на підприємстві Invers B.V. Практична цінність та наукове значення результатів роботи полягає в можливості застосування розроблених методик, моделей та прикладних програм для підтримки у процесі прийняття рішень, в тому числі у рамках процесу управління аутсорсинговими ІТ-проєктами.

Наукові результати дисертації використані у науково–дослідницькій діяльності Одеського національного політехнічного університету (довідка No 55/68-07 від 23.01.2023р.). Результати дисертації впроваджено у навчальний процес Одеського національного політехнічного університету МОН України, де їх використано при підготовці навчальних програм і методичного забезпечення дисциплін «Управління проєктами інформатизації» та «Експертні системи в формуванні стратегії розвитку підприємства» (довідка No 54/68-07 від 23.01.2023 р.).

Ключові слова: аутсорсингові ІТ-компанії, інноваційні технології, інформаційна економіка, ІТ-проєкт, інструментально-аналітичне забезпечення, управління ІТ-проєктами, обґрунтування економічних рішень, експертна оцінка, нечітка логіка, Stage-Gate фреймворк, FuzzyKIDE.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Dudnyk, O., Sokolovska, Z. (2022). Application of Fuzzy Expert Systems in IT Project Management. In M. Mircea, & T. M. M. Nguyen (Eds.), Project Management - New Trends and Applications. IntechOpen. https://doi.org/10.5772/intechopen.102439 *(0,62 д.а., особистий внесок здобувача: розроблено алгоритм оцінки виробничого потенціалу IT-компанії – 0,4 д.а.)*.
2. Дудник Олексій, Соколовська Зоя, "Devising a technology for managing outsourcing IT-projects with the application of fuzzy logic" // Eastern-European Journal Of Enterprise Technologies (Scopus), DOI: 10.15587/1729-4061.2021.224529, https://doi.org/10.15587/1729-4061.2021.224529, Indexing: Scopus, CrossRef, American Chemical Society, EBSCO, Index Copernicus, MIAR, CNKI, Ulrich's Periodical Directory, Neliti, 1findr, BASE, WorldCat, Polska Bibliografia Naukowa (*0,58 д.а., особистий внесок здобувача: розроблено модифікацію Stage-Gate фреймворку на основі нечіткої логіки – 0,45 д.а.*).
3. Дудник Олексій, Зоя Соколовська, "Forecasting development trends in the information technology industry using fuzzy logic" // Eastern-European Journal of Enterprise Technologies. 1, 74–85 (2023). (Scopus) https://doi.org/10.15587/1729-4061.2023.267906, Indexing: Scopus, CrossRef, American Chemical Society, EBSCO, Index Copernicus, MIAR, CNKI, Ulrich's Periodical Directory, Neliti, 1findr, BASE, WorldCat, Polska Bibliografia Naukowa (*0,5 д.а., особистий внесок здобувача: розроблено алгоритм прогнозування стану IT-індустрії – 0,35 д.а.*).
4. Дудник Олексій, Соколовська Зоя, "Modern Technologies for Management of Outsourcing IT Projects" // Business Inform (фаховий журнал), DOI: 10.32983/2222-4459-2021-2-62-72, https://doi.org/10.32983/2222-4459-2021-2-62-72 (*0,42 д.а., особистий внесок здобувача: проведено аналіз проблем та сучасних підходів до управління IT-проєктами – 0,3 д.а.*).

Наукові праці, які засвідчують апробацію матеріалів дисертації

5. Дудник Олексій, “Удосконалення процесів управління портфелем проєктів в IT-компаніях” // Економічна кібернетика: теорія, практика та напрямки розвитку (тези), https://economics.opu.ua/files/science/ek_kiber/2017/52.pdf *(0,12 д.а.).*
6. Дудник Олексій, “Аналіз оптимальної структури бази знань для нечітких експертних систем” // XI Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем” (тези, онлайн конференція - 11-12 квітня 2019 р), https://mpsesm.org/index.php/mpsesm/mpsesm-xi/paper/view/786/680 *(0,16 д.а.).*
7. Дудник Олексій, “Процес вибору виконавців IT-проєктів у портфоліо проєктів” // Економічна кібернетика: теорія, практика та напрямки розвитку” (тези), https://economics.opu.ua/files/science/ek_kiber/2018/50.pdf *(0,12 д.а.).*
8. Дудник Олексій, “Аналіз процесів управління портфелем проєктів в IT-компаніях” // X Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем” (тези, онлайн конференція - 5-6 квітня 2018 р), https://mpsesm.org/index.php/mpsesm/mpsesm-x/paper/view/670/571 *(0,16 д.а.).*
9. Дудник Олексій, “Переваги використання нечіткої логіки в процесі управління IT-проєктами” // XIII Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем” (тези, онлайн конференція - 8-9 квітня 2021 р), https://mpsesm.org/book/2021/thesis03-910.html *(0,16 д.а.).*
10. Дудник Олексій, “Економіка та апарат нечітких експертних систем: тренди розвитку та застосування” // Економічна кібернетика: теорія, практика та напрямки розвитку” (тези), https://economics.net.ua/wp-content/uploads/2023/01/tezy.pdf *(0,2 д.а.).*

Наукові праці, які додатково відображають наукові результати дисертації

11. Соколовська З.М., Дудник О.О. Експертні системи: теорія та прикладне застосування: [монографія]. Одеса: Екологія, 2022. 408 с. *(23,72 д.а., особистий внесок здобувача: розділи 3, 6, 7 – 10 д.а.).*

ABSTRACT

Dudnyk O.O. Innovative technologies in IT companies project portfolio management. – Qualification scientific work on the rights of the manuscript.

PhD degree dissertation – Doctor of Philosophy in specialty 051 – Economics. – Odessa Polytechnic National University Ministry of Education and Science of Ukraine, Odessa, 2023.

The dissertation is devoted to the application of innovative technologies in the process of managing outsourcing IT companies as instrumental and analytical support for effective management of IT projects and justification of economic decisions based on their expert assessment.

To date, there are a large number of outsourcing IT companies in Ukraine and the world, which are responsible for the development of a rather voluminous sector of IT projects. Project management processes within the outsourcing IT company are characterized by a number of features, the most important of which are increased responsibility for the software development sub-process, reduced or no responsibility for all other sub-processes, and partial financial dependence on the customer's company.

In this regard, there is a growing interest in innovative approaches in the development of instrumental and analytical support for effective management of IT projects and justification of economic decisions based on an expert assessment of the current status of these projects, which in turn increases the efficiency of the software development sub-process, allowing to minimize negative consequences financial dependence on the customer's company.

Chapter 1 deals with the theoretical principles of managing IT projects and their portfolios. The importance of economic indicators and project success metrics for achieving success in project management is demonstrated. Attention is focused on the importance of taking into account not only technical, but also business aspects of the IT project.

An analysis of the main trends of the information economy was carried out. It has been established that the global trends are the growth of the volume of data, the development of artificial intelligence technologies and the Internet of Things. Ukraine is no exception and is also actively developing in the field of IT. The role of the information economy in Ukraine is very important, as this industry is a source of increasing GDP and accelerating economic development. Tensions on the global stage, including the Russian-Ukrainian war and the COVID-19 pandemic, also have a significant impact on Ukraine's information economy. This has become the cause of serious challenges for business, in particular for IT companies. For example, a drop in demand for services and products, or a lack of funding can lead to staff cuts and limited opportunities to implement new IT projects. In the light of these factors, effective technologies and IT project management processes, which allow quick response to changes and minimize risks, become important.

The main problems and risks of project management in IT companies have been identified. On the one hand, the rapid development of technologies can be useful for IT companies, because new tools and technologies ensure more efficient execution of projects and reduce costs for software development. However, on the other hand, constant changes create problems and risks in project management: for example, changes in market conditions, technical problems, insufficient qualification of personnel, improper planning of resources and many other factors. In addition, the information economy has an impact on market conditions and industry competition. Modern technologies allow IT companies to speed up software development, which creates tougher competition and can affect the price of IT services, as innovative solutions usually cost more than traditional ones. Thus, in case of ineffective management of IT projects, there may be a delay in their implementation, an increase in costs, or even a complete failure of the projects.

The main IT project management technologies have been studied. Among them, Agile technology and Stage-Gate framework are highlighted. Agile technology is based on iterative incremental development of a software product and active participation of the customer in the development process. The advantage of using this

technology is a quick response to changes in customer requirements and a reduction in the time from development to the release of the finished product. However, its disadvantages are the need for a high level of communication and coordination between members of the development team and the customer, which can lead to delays in the implementation of the IT project. The Stage-Gate framework, developed by Robert Cooper, is based on a sequential passage through the stages of product development, with the establishment of decision points for moving to the next stages. Its advantages are a more structured and predictable process of software development, which allows you to more accurately determine the results of each stage. However, the disadvantages are a more complex decision-making procedure at each stage, which can lead to delays in the implementation of the project.

The research results presented in Chapter 1 allow us to conclude that the use of fuzzy logic to solve IT project management tasks is a rather rare phenomenon. More attention is paid to the process in general, building various frameworks aimed at the general description of the process, as well as its fragmentation into smaller tasks. But the procedures for solving these problems are much less studied by researchers. There are only a few studies on process subtasks using fuzzy logic; there are almost no precedents for using fuzzy logic in relation to the process as a whole. The use of fuzzy logic for solving IT project management problems can be effective, because fuzzy logic allows taking into account not only the exact values of parameters, but also their fuzzy values, which can improve the degree of validity and the overall results of economic decision-making; simplify the process of their implementation.

In the conditions of growing competition on the market, the task of managing IT projects is becoming more and more difficult. Managing a portfolio of IT projects requires understanding a fairly large number of factors that affect the effectiveness of software design and development. Instrumental and analytical support is an extremely important tool for managing IT projects. Using the apparatus of fuzzy logic makes it possible to take into account the vagueness of data and knowledge, which is important in the conditions of instability of the management process and a significant degree of uncertainty in the interaction between project participants.

In this regard, chapter 2 proposes a conceptual model of substantiation and making economic decisions on project management in the conditions of an uncertain environment and inaccurate information. The basic theoretical foundations of fuzzy logic as a mathematical apparatus designed for working with vague, ambiguous, partial and incomplete data are considered. The concepts of fuzzy set, membership functions and fuzzy operator are considered. The main advantages of fuzzy intelligent systems implemented on the basis of the mathematical apparatus of fuzzy logic are highlighted.

The applied instrumental and analytical base for the implementation of the economic decision-making model is the FuzzyKIDE fuzzy expert system: the author's software shell, implemented on the basis of fuzzy logic. FuzzyKIDE architecture is based on data fuzzification, which allows representing each element in the database as a function of belonging to some set from the knowledge base. Compared to other fuzzy expert systems, FuzzyKIDE uses an optimized knowledge base model. The system's knowledge about the object of functioning under investigation is described by fuzzy inference rules, linguistic variables, and membership functions. Each function describes to what extent an element corresponds to a certain characteristic, that is, it describes the degree of belonging to some fuzzy set. Fuzzy rules define relationships between input and output linguistic variables. This approach makes it possible to easily modify the knowledge base, add new rules or make changes to existing ones. Another feature is the use of SNEPS semantic networks in the core of fuzzy inference, which allows you to reduce the number of calls to the knowledge base and database and increase the efficiency of using the knowledge of the system. Thus, FuzzyKIDE has many advantages compared to classical expert systems, in particular, a structured model of the knowledge base and the possibility of using semantic networks for optimal storage and use of knowledge.

A modification of the Agile Stage-Gate framework using fuzzy systems is proposed, which allows taking into account the features of IT projects as a whole and the features of modern iterative development cycles. A feature of the proposed scheme is the use of fuzzy logic systems when moving from the gates to the next

stages. There is one significant feature in the way the gates are placed between the stages - some of the gates are not unidirectional. For example, the gate after the project search process is unidirectional, because the project search process itself is not iterative. At the same time, the gate before the Agile sprint process is two-way, because after each sprint there is a process of evaluating the results at the same gate in order to determine whether there is a need to go back to the previous stage.

The mathematical apparatus proposed in the section is aimed at solving a wide range of problems related to various aspects of the justification of strategic and operational economic decisions, which require a comprehensive analysis of the influence of factors of various nature; taking into account entropy and incompleteness/inaccuracy of information regarding the internal and external environment of the studied processes.

The efficiency of the developed instrumental and analytical decision-making base is demonstrated in chapter 3 on the results obtained during the solution of typical tasks of IT project management in outsourcing companies: project search and formation of the composition and structure of the portfolio; monitoring of the current status of projects; strategic orientation of the IT company (professional orientation and profit maximization). A comparative analysis of available historical data and data obtained as a result of the operation of the FuzzyKIDE software shell, implemented in the outline of the modified Agile Stage-Gate framework, proved the effectiveness of the proposed approach. The comparison showed that the use of the software shell significantly increases the efficiency of the decision-making process and leads to a qualitative improvement of the absolute and relative economic indicators of the activity of the outsourcing IT company.

Further exploitation of the developed mathematical apparatus will lead to the enrichment of knowledge and data bases, which will contribute to the development of information support for the processes of supporting the adoption of sound economic decisions regarding the management of IT projects/portfolio projects.

The results of the dissertation work were implemented at the Invers B.V. enterprise. The practical value and scientific significance of the work results lies in

the possibility of applying the developed methods, models and application programs to support the decision-making process, including within the framework of the outsourcing IT project management process.

The scientific results of the dissertation were used in the scientific and research activities of the Odessa National Polytechnic University (certificate No. 55/68-07 dated January 23, 2023). The results of the dissertation were implemented in the educational process of the Odesa National Polytechnic University of the Ministry of Education and Culture of Ukraine, where they were used in the preparation of educational programs and methodological support for the disciplines "Management of informatization projects" and "Expert systems in the formation of enterprise development strategy" (reference No. 54/68-07 dated 23.01. 2023).

Keywords: outsourcing IT companies, innovative technologies, information economy, IT project, instrumental and analytical support, IT project management, justification of economic decisions, expert assessment, fuzzy logic, Stage-Gate framework, FuzzyKIDE.

LIST OF THE APPLICANT'S PUBLICATION

Scientific works in which the main scientific results of the dissertation are published

1. Dudnyk, O., Sokolovska, Z. (2022), "Application of Fuzzy Expert Systems in IT Project Management. In M. Mircea, & T. M. M. Nguyen (Eds.), Project Management - New Trends and Applications" IntechOpen. https://doi.org/10.5772/intechopen.102439 (0.62 p.p., the personal contribution of the acquirer: an algorithm for assessing the production potential of the IT company was developed - 0.4 p.p.).
2. Dudnyk, O., Sokolovska, Z., "Devising a technology for managing outsourcing IT-projects with the application of fuzzy logic" // Eastern-European Journal Of Enterprise Technologies (Scopus), DOI: 10.15587/1729-4061.2021.224529, https://doi.org/10.15587/1729-4061.2021.224529, Indexing: Scopus, CrossRef, American Chemical Society, EBSCO, Index Copernicus, MIAR, CNKI, Ulrich's Periodical Directory, Neliti, 1findr, BASE, WorldCat, Polska Bibliografia Naukowa (0.58 p.p., personal contribution of the acquirer: Stage-Gate modification of the framework based on fuzzy logic was developed - 0.45 p.p.).
3. Dudnyk, O., Sokolovska, Z., "Forecasting development trends in the information technolo-gy industry using fuzzy logic" // Eastern-European Journal of Enterprise Technologies. 1, 74–85 (2023). https://doi.org/10.15587/1729-4061.2023.267906, Indexing: Scopus, CrossRef, American Chemical Society, EBSCO, Index Copernicus, MIAR, CNKI, Ulrich's Periodical Directory, Neliti, 1findr, BASE, WorldCat, Polska Bibliografia Naukowa (0.5 p.p, personal contribution of the acquirer: an algorithm for forecasting the state of the IT industry was developed - 0.35 p.p.).
4. Dudnyk, O., Sokolovska, Z., "Modern Technologies for Management of Outsourcing IT Projects" // Business Inform (professional journal), DOI: 10.32983/2222-4459-2021-2-62-72, https://doi.org/10.32983/2222-4459-2021-2-62-72 (0.42 p.p, personal contribution of the acquirer: analysis of problems and modern approaches to IT project management was carried out - 0.3 p.p.).

Scientific works certifying the approbation of the dissertation materials

5. Dudnyk, O., “Improvement of project portfolio management processes in IT companies” // Economic cybernetics: theory, practice and directions of development (theses), https://economics.opu.ua/files/science/ek_kiber/2017/52.pdf *(0,12 p.p.).*
6. Dudnyk, O., “Analysis of the optimal structure of the knowledge base for fuzzy expert systems” // XI International Scientific and Practical Internet Conference "Modern Problems of Modeling Socio-Economic Systems" (theses, online conference - 11-12 April 2019), https://mpsesm.org/index.php/mpsesm/mpsesm-xi/paper/view/786/680 *(0,16 p.p.).*
7. Dudnyk, O., “The process of selecting executors of IT projects in the project portfolio” // Economic cybernetics: theory, practice and directions of development (theses), https://economics.opu.ua/files/science/ek_kiber/2018/50.pdf *(0,12 p.p.).*
8. Dudnyk, O., “Analysis of project portfolio management processes in IT companies” // X International Scientific and Practical Internet Conference "Modern Problems of Modeling Socio-Economic Systems" (theses, online conference - 5-6 April 2018), https://mpsesm.org/index.php/mpsesm/mpsesm-x/paper/view/670/57 *(0,16 p.p.).*
9. Dudnyk, O., “Advantages of using fuzzy logic in the IT project management process” // XIII International Scientific and Practical Internet Conference "Modern Problems of Modeling Socio-Economic Systems" (theses - 8-9 April 2021), https://mpsesm.org/book/2021/thesis03-910.html *(0,16 p.p.).*
10. Dudnyk, O., “Economics and apparatus of fuzzy expert systems: development trends and applications” // Economic cybernetics: theory, practice and directions of development (theses), https://economics.net.ua/wp-content/uploads/2023/01/tezy.pdf *(0,2 p.p.).*

Scientific works that additionally reflect the scientific results of the dissertation

11. Sokolovska, Z., Dudnyk, O., Expert systems: theory and practical application: [monograph]. Odesa: Ecology, 2022. 408 c. (23.72 p.p., personal contribution of the acquirer: sections 3, 6, 7 - 10 p.p.).

ЗМІСТ

ВСТУП……. 20
РОЗДІЛ 1. ТЕОРЕТИЧНІ ЗАСАДИ ЗАСТОСУВАННЯ ІННОВАЦІЙНИХ ТЕХНОЛОГІЙ В ПРОЦЕСІ УПРАВЛІННЯ ПОРТФЕЛЕМ ПРОЄКТІВ В ІТ-СФЕРІ……..... 29
1.1 Основні тренди та роль інформаційної економіки в Україні та світі……..... 29
1.2 Ключові аспекти та класифікація проблем управління проєктами ІТ-компаній…… 45
1.3 Сучасні інноваційні технології управління ІТ-проєктами: зміст та етапи застосування…….. 64
Висновки до розділу 1…… 94
РОЗДІЛ 2. ІНСТРУМЕНТАЛЬНО-АНАЛІТИЧНЕ ЗАБЕЗПЕЧЕННЯ УПРАВЛІННЯ АУТСОРСИНГОВИМИ ІТ-ПРОЄКТАМИ: АПАРАТ НЕЧІТКОЇ ЛОГІКИ В КОНТУРІ УПРАВЛІННЯ ПОРТФОЛІО ПРОЄКТІВ…… 97
1.1 Робота з нечіткими знаннями як основа формування іструментально-аналітичного забезпечення управління аутсорсинговими ІТ-проєктами……. 97
1.2 Концептуальна платформа розробки нечіткої експертної системи FuzzyKIDE: архітектура,модельна основа та механізм логічного виводу……104
2.2.1 Архітектура типової експертної системи……..... 104
2.2.2 Модель побудови бази знань FuzzyKIDE……..... 110
2.2.3 Механізм логічного виводу FuzzyKIDE…….. 113
2.2.4 Архітектура оболонки FuzzyKIDE…… 124
1.3 Методичні засади та технологія роботи з нечіткою експертною системою FuzzyKIDE…… 129
2.4. Модель Agile Stage-Gate фреймворку з використанням

нечіткої логіки: зміст, функціонал та переваги…… 138
Висновки до розділу 2……. 150
РОЗДІЛ 3. ЕКСПЕРТНІ КОНСУЛЬТАЦІЇ З FUZZYKIDE ЯК АНАЛІТИЧНА ОСНОВА ОБГРУНТУВАННЯ ТА ПРИЙНЯТТЯ ЕКОНОМІЧНИХ РІШЕНЬ З УПРАВЛІННЯ ПОРТФОЛІО ПРОЄКТІВ..152
3.1 Розв’язання задач стратегічного управління ІТ-компанією в умовах невизначеного середовища……. 152
3.1.1 Стратегічна орієнтація компанії: професійне спрямування…….. 153
3.1.2 Стратегічна орієнтація компанії: максимізація прибутку……. 166
3.2 Розробка економічних рішень з формування портфоліо проєктів……. 183
3.3 Методи моніторингу поточного статусу проєктів/портфоліо проєктів…… 192
3.4 Аналіз результатів впровадження та обгрунтування ефективності застосування інноваційної технології – експертної системи FuzzyKIDE – в ІТ-сфері…… 202
Висновки до розділу 3……. 218
ВИСНОВКИ…….. 220
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ……. 225
ДОДАТОК А. Список публікацій здобувача…… 237
ДОДАТОК Б. Відомості про апробацію результатів дисертації…….. 239
ДОДАТОК В. Довідка про використання результатів дисертаційної роботи у науково-дослідницької діяльності…… 240
ДОДАТОК Г. Довідка про впровадження результатів дисертаційної роботи у навчальний процес……... 242

ВСТУП

Актуальність теми. Сьогодні більшість IT-компаній відповідають за розробку великої кількості проєктів одночасно. Управління проєктами розробки програмного забезпечення на даний момент стало великим полем як наукової, так і практичної діяльності. В області виробництва програмного забезпечення задіяно понад два мільйони фахівців по всьому світу, 20% яких становлять менеджери. Вони змушені контролювати протікання проєкту в умовах постійних змін. Саме тому дослідники процесу управління проєктами в першу чергу намагаються виділити загальні процеси, присутні у кожному проєкті. Так як процес управління численними проєктами складається з поодиноких проєктів, то їх загальні обмеження також дуже важливі. Але треба розуміти, що ситуація стає більш складною зі збільшенням кількості проєктів. При управлінні численними проєктами необхідно збалансувати часто суперечливі вимоги з обмеженими ресурсами, координувати націленість проєктів з метою забезпечення оптимального загального результату. Таким чином, питання управління кількома проєктами створює новий набір задач, котрі компанії необхідно вирішити.

Аналіз останніх досліджень в галузі управління портфелями IT-проєктів показав, що ефективне управління ними є основним способом їх успішного завершення в рамках обмежень часу, вартості та якості [32, 34, 90]. За даними різних дослідницьких груп CompTIA, станом на кінець 2021 року, з 5000 відстежених портфелів програмних проєктів через неефективне управління 23% завершилися невдало, 36% зажадали додаткових витрат часу і фінансів і тільки 41% успішно завершився [60]. За результатами того ж дослідження, в середньому бюджет портфелів проєктів перевищується на 62%, а витрачений час на 22% перевищує оцінений. При цьому реалізується в середньому всього 85% заявленої в специфікації функціональності. Більшість методів орієнтується на розв’язання окремих задач управління портфелем, в першу чергу на його

формування і оптимізацію. В реальності портфель, як об'єкт управління, має динамічний характер, і управління ним повинно бути циклічним, що передбачає постійний моніторинг результатів його реалізації і, відповідно, регулярне переформування портфеля. Тому актуальною є розробка і дослідження моделей і методів ефективного управління портфелями проєктів − набором технологічно незалежних проєктів, що реалізуються організацією в умовах ресурсних обмежень для досягнення її стратегічних цілей. При цьому, міра досягнення стратегічних цілей компанії є показником ефективного управління портфелем проєктів.

Проблемам управління портфелем проєктів присвячена значна кількість літературних джерел, серед яких [30, 34, 90, 107]. Достатньо визначена і традиційна інструментальна база процесів управління [26, 39, 40, 69, 91, 94, 101].

Однак, задача ефективного управління портфелем проєктів належить до завдань оптимізації в умовах невизначеності. Як правило, для розв’язання подібних завдань залучається апарат теорії ймовірності. Водночас, в ряді ситуацій застосування теорії ймовірностей представляється недостатньо коректним і обґрунтованим. Причиною цього є нестача наявних даних, що не дозволяє з достатнім ступенем впевненості встановити адекватність обраної для опису ситуації ймовірнісної моделі.

В таких умовах з'являється потреба в інших, відмінних від ймовірного, підходах до оцінки наявної невизначеності. Один з таких підходів заснований на застосуванні теорії нечітких множин.

Не зважаючи на те, що апарат теорії нечітких множин є не новим і визначеним у значному колі спеціальних наукових праць ([24, 48, 56, 70, 78]), його прикладне застосування все ще є достатньо обмеженим. В повній мірі це стосується проблеми управління портфелем IT проєктів. Потребує окремого дослідження і взаємодія даного математичного апарату з прийнятими спеціальними технологіями управління портфоліо в IT галузі.

Все це призводить до необхідності розробки й прикладного впровадження інноваційних технологій, які б поєднали апарат нечіткої логіки з сучасними підходами до управління ІТ-проєктами.

Зазначене зумовило вибір теми, мету, завдання, предмет, структуру та напрями дослідження.

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

Дисертаційну роботу виконано у відповідності до планів науково-дослідних робіт Національного університету «Одеська політехніка» за 2017-2023 рр. при виконанні наступних держбюджетних тем: № 119-68 «Сучасні технології аналізу та моделювання динаміки розвитку соціально-економічних процесів на різних рівнях ієрархії управління» (номер державної реєстрації 0116U004541, 2016-2020 рр.), де автором здійснено аналіз сучасного стану та трендів розвитку світової та вітчизняної ІТ індустрії; представлено основні теоретичні засади управління портфелем проєктів в ІТ сфері; доведено доцільність використання нечіткої логіки в контурі управління портфоліо проєктів; наведено архітектуру нечіткої експертної системи FuzzyKIDE (оригінальної авторської розробки) та основні математичні нотації її побудови; доведено відмінності розробленої системи від існуючих зразків нечітких експертних оболонок; № 195-68 «Теоретичні і прикладні проблеми системного аналізу, сучасних технологій і моделювання в соціально-економічному просторі» (номер реєстрації 0121U108286, 2021-2025 рр.), де автором представлена технологія роботи з нечіткою експертною системою FuzzyKIDE; наведені результати експертних консультацій, здійснених в процесі розв'язання низки задач з управління портфелем ІТ проєктів; доведено значення використання запропонованих інтелектуальних технологій як аналітичної основи прийняття рішень з управління портфоліо ІТ проєктів.

Дисертант приймав участь в НДР як співвиконавець.

Мета і задачі дослідження. Метою роботи є розробка інноваційної основи інструментально-аналітичного забезпечення ефективного управління

аутсорсинговими ІТ-проєктами та обґрунтування економічних рішень на основі їх експертного оцінювання.

Досягнення поставленої мети передбачає вирішення наступних *завдань дослідження*:

− провести дослідження динаміки і особливостей розвитку вітчизняної та світової ІТ-галузі із визначенням типових проблем, що виникають в процесах управління проєктами та портфелями проєктів ІТ-компаній;

− запропонувати концептуальну модель обґрунтування й прийняття економічних рішень з управління аутсорсингових ІТ-проєктів в умовах невизначеного середовища і неточної інформації;

− розробити прикладну інструментальну-аналітичну базу підтримки прийняття рішень з управління ІТ-проєктами/портфелями проєктів;

− дослідити процеси розв’язання задач стратегічного управління ІТ-компанією в умовах невизначеного середовища;

− дослідити процеси розробки ІТ-компанією економічних рішень з формування портфоліо проєктів;

− проаналізувати методи моніторингу поточного статусу ІТ-проєктів /портфоліо проєктів;

− розвинути інформаційне забезпечення процесів розробки економічних рішень з управління аутсорсинговими ІТ-проєктами.

Об'єктом наукового дослідження є процеси управління аутсорсинговими ІТ-проєктами.

Предметом наукового дослідження є інноваційні технології створення інструментально-аналітичної бази обґрунтування й прийняття економічних рішень з управління аутсорсинговими ІТ-проєктами.

Методи дослідження. Для вирішення поставлених завдань було використано такі загальнонаукові та спеціальні методи: структурно-логічного аналізу – для побудови структури та узагальнення теоретичних та практичних положень дослідження; системного аналізу та абстрактно-логічного

узагальнення – при дослідженні основ управління проєктами IT-аутсорсингових компаній; табличний та графічний методи – для наочного подання отриманих результатів та аналітичних даних; ідеї, принципи та підходи застосування нечітких множин, нечіткої логіки, лінгвістичних змінних та інших понять теорії нечітких множин в ході розробки концептуальної моделі системи управління IT-проєктами; Agile та фреймворк технології управління IT-проєктами, методи інтелектуального аналізу даних, математичні інструменти та технології проектування оболонок експертних систем в ході розробки інструментально-аналітичної бази підтримки прийняття рішень з управління проєктами – нечіткої експертної системи FuzzyKIDE.

Інформаційну базу дослідження склали праці вітчизняних і зарубіжних вчених, фахівців-практиків з проблематики дослідження; міжнародні стандарти, чинне законодавство у сфері інформаційних технологій; статистичні дані Державної служби статистики України; публічні огляди та відкриті дані стосовно функціонування вітчизняної та світової IT-галузі; дані, розміщені на офіційних веб-сайтах IT-компаній; матеріали IT-компаній HYS Enterprise B.V. та INVERS B.V.; результати власних досліджень.

Наукова новизна одержаних результатів полягає в розробленні інструментально-аналітичного забезпечення ефективного управління IT-проєктами та обґрунтування економічних рішень на основі їх експертного оцінювання, а саме:

− вперше розроблено концептуальну модель системи управління аутсорсинговими IT-проєктами, яка базується на використанні нечіткої логіки в загальному контурі прийняття рішень, що забезпечує врахування особливостей реалізації проєктів в умовах невизначеності та неповноти/неточності інформації. Загальний контур управління проєктами реалізований в нотаціях Stage-Gate фреймворку, де перехід між окремими стадіями (подолання відповідних брам) передбачає підключення нечіткої системи прийняття рішень. Згідно базовій концепції модель має забезпечувати: а) можливість визначення правил прийняття рішень і правил переходу з етапу на етап у лексичному

форматі; б) зменшення залежності від значного обсягу точних чисельних даних; в) підвищення прозорості процесу просування проєкту; г) підвищення ефективності управління в результаті оперативної обробки зовнішніх змін, специфічних для реалізації ІТ-проєктів;

− вперше розроблено прикладну інструментально-аналітичну базу підтримки прийняття рішень з управління проєктами – нечітку експертну систему FuzzyKIDE. Оптимізована модель бази знань експертної системи поєднує модифіковані нечіткі продукційні правила та спрощені лінгвістичні змінні, що дозволяє позбутися бази проміжних фактів та використати семантичну мережу для побудови модифікованого механізму нечіткого виводу. Запропонована архітектура з наявною метакомпонентою та простим користувальницьким інтерфейсом дозволяє: а)зменшити навантаження на базу даних та прискорити процеси нечіткого виводу; б)забезпечити модульну гнучку структуру бази знань та даних; в)забезпечити зручну технологію підтримки бази даних та знань в актуальному стані; г)реалізувати оперативні та змістовні консультації з розробки й прийняття економічних рішень щодо управління ІТ-проєктами; д)отримувати аналітичні довідки системи стосовно кінцевих результатів консультацій;

удосконалено:

− методичний підхід до розв’язання задач стратегічного управління аутсорсинговою ІТ-компанією: а) стратегічної орієнтації компанії (професійне спрямування), що відрізняється обґрунтуванням цілей й моніторингу їх дотримання шляхом здійснення комплексного (як кількісного, так і якісного) аналізу на базі математичного інструментарію FuzzyKIDE; б) стратегічної орієнтації компанії (максимізація прибутку), що відрізняється прогнозуванням доходної та витратної частин реалізації ІТ-проєктів на базі відтворення різноманітних модельних ситуацій аутсорсингових ІТ-компаній;

− методичний підхід до обґрунтування й прийняття економічних рішень стосовно формування аутсорсинговою ІТ-компанією складу і структури портфоліо проєктів у поточному періоді та на перспективу, що на відміну від

існуючих реалізується за рахунок удосконалення комунікацій і суттєвого підвищення вхідного інформаційного навантаження процесу шляхом використання баз даних та знань експертної системи FuzzyKIDE;

− методи моніторингу поточного статусу ІТ-проєктів/портфоліо проєктів, які відрізняються здійсненням багатофакторного аналізу впливів внутрішнього та зовнішнього оточення аутсорсингових ІТ-компаній з використанням апарату нечіткої логіки, що призводить до підвищення оперативності та гнучкості прийнятих управлінських рішень;

дістало подальшого розвитку:

− інформаційне забезпечення процесів прийняття економічних рішень з управління аутсорсинговими ІТ-проєктами/портфоліо проєктів, що на відміну від існуючої інформаційної підтримки досягається за рахунок створення та актуалізації баз даних і знань нечіткої експертної системи FuzzyKIDE в процесі польових емпіричних випробувань.

Практичне значення одержаних результатів полягає в розробці науково-обґрунтованого інструментально-аналітичного забезпечення ефективного управління проєктами аутсорсинговими ІТ-компаніями. Результати роботи і рекомендації впроваджено ІТ компанією INVERS B.V. (довідка від 01.09.2022 р.).

Результати використані у науково-дослідницькій діяльності Національного університету «Одеська політехніка» (довідка № 55/68-07 від 23.01.2023 р.).

Результати дисертації впроваджено у навчальний процес Національного університету «Одеська політехніка» МОН України, де їх використано при підготовці навчальних програм, навчально-методичних матеріалах та курсів лекцій з дисциплін «Управління проєктами інформатизації» і «Експертні системи в формуванні стратегії розвитку підприємств» (довідка № 54/68-07 від 23.01.2023 р.).

Особистий внесок здобувача. Дисертація є самостійно виконаною науковою працею, в якій викладено авторський підхід до застосування

інноваційних технологій в управлінні портфелем проєктів ІТ-компаній, створення інструментально-аналітичного забезпечення обґрунтування й прийняття економічних рішень з управління аутсорсинговими ІТ-проєктами. Всі наукові результати, викладені в дисертації, одержано автором особисто. З наукових публікацій, виданих в співавторстві, у роботі використані лише ті положення, що складають індивідуальний внесок, який зазначений у переліку публікацій за темою дисертації.

Апробація результатів дослідження. Теоретико-методологічні положення, прикладні результати і рекомендації дисертації оприлюднювались та обговорювались на міжнародних науково-практичних конференціях: XIII Міжнародна науково-практична Інтернет-конференція "Сучасні проблеми моделювання соціально-економічних систем" (м. Харків, 2018 р.), Міжнародна науково-практична конференція "Економічна кібернетика: теорія, практика та напрямки розвитку" (м. Одеса, 2022 р.), X Міжнародна науково-практична Інтернет-конференція "Сучасні проблеми моделювання соціально-економічних систем" (м. Харків, 2021 р.), XI Міжнародна науково-практична Інтернет-конференція "Сучасні проблеми моделювання соціально-економічних систем" (м. Харків, 2019 р.), Міжнародна науково-практична конференція "Економічна кібернетика: теорія, практика та напрямки розвитку" (м. Одеса, 2018 р.), Міжнародна науково-практична конференція "Економічна кібернетика: теорія, практика та напрямки розвитку" (м. Одеса, 2017 р.).

Публікації. За результатами досліджень опубліковано 11 наукових праць: 1 монографія (в співавторстві); 1 стаття у науковому періодичному виданні іншої держави; 3 статті у наукових фахових виданнях України, що включені у міжнародні наукометричні бази (2 статті – Scopus); 6 – тез доповідей на наукових конференціях. Загальний обсяг публікацій складає 26,74 д.а., з яких 12,24 д.а. належать особисто автору.

Структура та обсяг дисертації. Дисертаційна робота складається з анотації, вступу, трьох розділів, висновків, списку використаних джерел з 126 найменувань – на 12 сторінках, 4 додатків на 6 сторінках. Повний обсяг

дисертації − 242 сторінки, в тому числі 205 сторінок основного тексту. Дисертація містить 70 рисунків, з яких 3 займають повні сторінки; 33 таблиці, з яких 2 займають повні сторінки.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ЗАСАДИ ЗАСТОСУВАННЯ ІННОВАЦІЙНИХ ТЕХНОЛОГІЙ В ПРОЦЕСІ УПРАВЛІННЯ ПОРТФЕЛЕМ ПРОЄКТІВ В ІТ-СФЕРІ

1.1 Основні тренди та роль інформаційної економіки в Україні та світі

Розвиток людства в останні десятиліття супроводжувався швидкими змінами в технологіях і зростаючим поширенням оцифрованих пристроїв і послуг. Це супроводжується розширенням аналітики великих даних, штучного інтелекту, хмарних обчислень та цифрових платформ. Пандемія COVID-19 прискорила процеси цифровізації, оскільки все більше і більше людей продовжують, наскільки це можливо, свою діяльність через онлайн-канали – наприклад, для роботи, навчання, спілкування, продажу та купівлі чи розваг. Через це роль цифрових даних та технологій все більш поширюється, а цифрова економіка розвивається з шаленою швидкістю. Індустрія інформаційно-комунікаційних технологій (ІКТ) стоїть у центрі більшої частини цієї діяльності, підтримуючи цифрову економіку та слугуючи надійним критерієм ефективності.

ІКТ відіграють вирішальну роль в економіці не тільки як джерело потенційного доходу, але і як вектор перехресного зростання, вносячи глибокі зміни в різні сектори економіки. Такі технології як інтернет речей, робототехніка, штучний інтелект, хмарні обчислення, аналіз великих даних, 3D друк та багато інших вже змінюють спосіб проєктування фабрик, виготовлення продуктів та надання послуг. Те саме стосується всіх галузей економіки, включаючи сільське господарство, де інтернет речей та робототехніка матимуть дедалі більший вплив, особливо на робоче місце та навички, які користуються попитом.

Трансформаційний вплив оцифрування та інформаційних технологій стає найбільш очевидним, якщо врахувати зростаючу важливість великих технологічних компаній та цифрових платформ за останнє десятиліття. Про різкі зміни свідчить порівняння секторів 100 найбільших компаній. У 2009 році найбільшим сектором був нафтогазовий, представлений 18 компаніями з топ-100, тоді як з сектору технологій та побутових послуг був лише на п'ятому місці з 11 компаніями. До 2022 року картина суттєво змінилася: кількість технологічних компаній у топ-100 зросла до 20 і цей сектор вийшов на перше місце. Натомість з нафти, газу та видобутку корисних копалин залишилась лише шість компаній – табл. 1.1. [87, 88].

Водночас, з 10 найбільших компаній 2009 року лише одна залишилась в топ-10 2022 року. П'ять з 10 найкращих фірм у 2022 році навіть не представлені серед 100 найкращих у 2009 році: Amazon, Meta, Tesla, TSMC та Tencent [87, 88].

Зміна ще більш помітна, якщо вимірювати її за ринковою капіталізацією. У 2009 році на підприємства нафтового та газового секторів припадало 23 відсотки загальної ринкової капіталізації топ-100 компаній, за ними йшли фінансові послуги з часткою 16 відсотків, тоді як технологічні та побутові послуги представляли 12 відсотків. До 2022 року частка останніх зросла до 34 відсотків, частка фінансових послуг у загальній ринковій капіталізації значно зменшилась до 11 відсотків, а енергетичного сектора – до 10 відсотків – табл. 1.1 [112].

Найбільші цифрові компанії світу дуже сконцентровані географічно в Америці та Азії, а решта світу, особливо країни Африки, значно відстають – рис. 1.1 [46, 113]. З точки зору спроможності брати участь у цифровій економіці і отримувати користь від неї виділяються дві країни: Сполучені Штати та Китай. Разом на них припадає половина світових гіпермасштабних центрів обробки даних, найвищі показники впровадження 5G у світі, 94 відсотки всього фінансування стартапів штучного інтелекту за останні п'ять років, 70 відсотків

найкращих світових дослідників штучного інтелекту та майже 90 відсотків ринкової капіталізації найбільших у світі цифрових платформ [112].

Аналіз даних веб-трафіку також підтверджує домінування цифрових платформ США. Сполучені Штати розміщують більше половини із 100 найкращих веб-сайтів, що використовуються в 9 із 13 субрегіонів світу. Навіть у Західній Європі веб-сайти, які найчастіше використовуються, базуються в США [77].

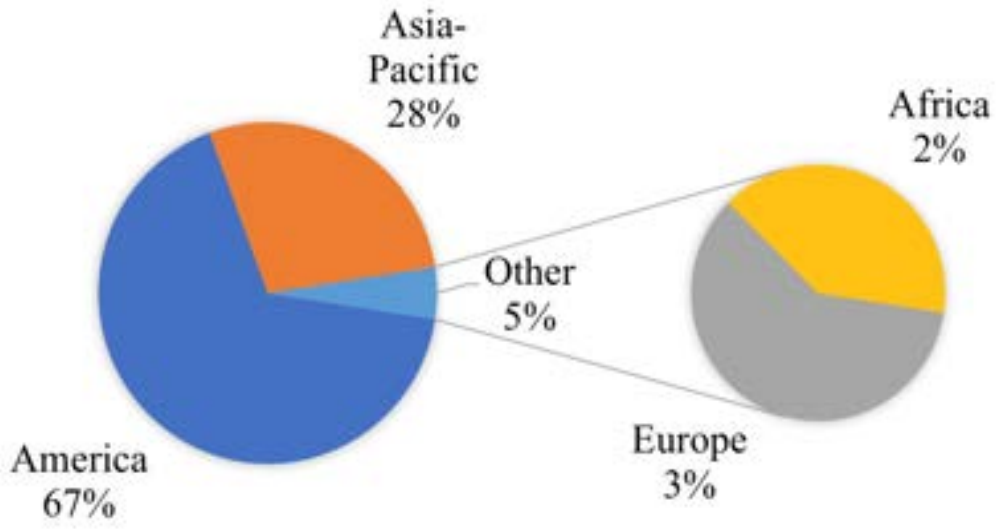


Рисунок 1.1 – Регіональний розподіл 100 найкращих світових платформ у світі станом на 2020 (Джерело – [власна розробка])

Внаслідок прискорених темпів технологічних інновацій, наразі IT сектор є набагато більшим, ніж був п'ять років тому. Прогнозується, що у 2022 році загальносвітові витрати на IT становитимуть 4,5 трлн дол. США, що на 30% більше, ніж у 2016 році [51, 53]. Щорічні темпи приросту світової IT-галузі до пандемії, в період 2016-2019 рр., знаходились у стабільному діапазоні від 2% до 5%. Під час пандемії ІКТ сектор, як і всі інші галузі економіки, зазнав кризи, але значно виріс у 2021 році – рис. 1.2 [52].

Таблиця 1.1 – Порівняння секторів топ 100 компаній за ринковою капіталізацією у 2009 та 2022 роках (Джерело – [власна розробка])

Топ 100 2022					Топ 100 2009		
Сектор	Зміна у ринковій капіталізації ($млрд.)	Ранг	Ринкова капіталізація ($млрд.)	Кільк. компаній	Ранг	Ринкова капіталізація ($млрд.)	Кільк. компаній
Технології	10 994	1	11 991	20	5	997	11
Охорона здоров’я	3 014	2	4 098	16	4	1 084	14
Фінанси	2 638	3	3 964	18	2	1 326	17
Енергія	1 631	4	3 560	6	1	1 929	18

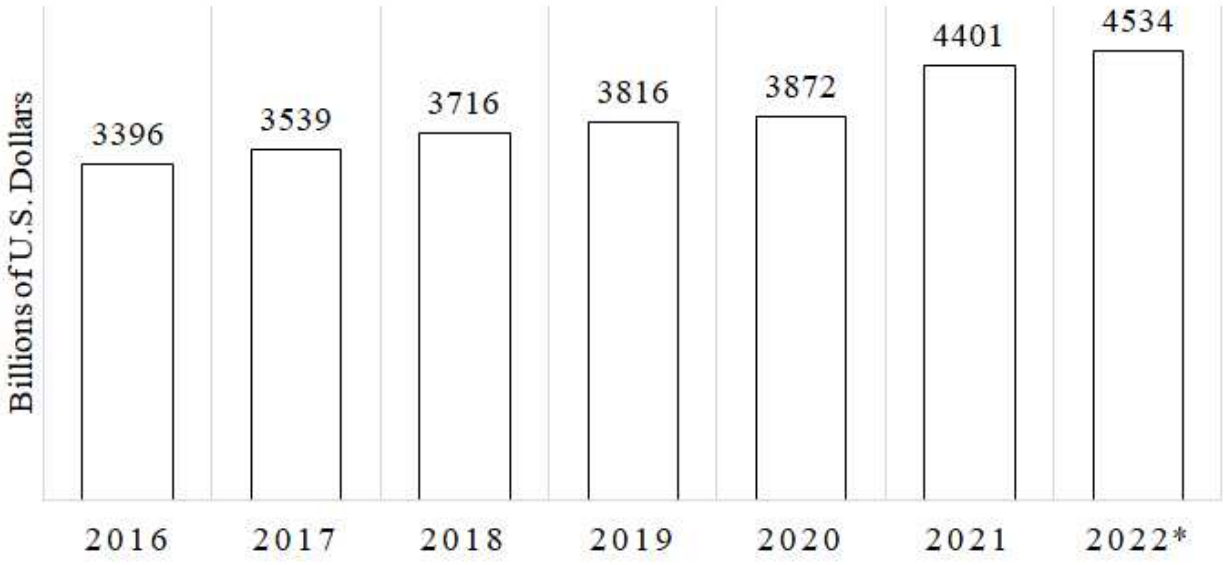


Рисунок 1.2 – Динаміка витрат на IT у 2016-2022 роках

(Джерело – [власна розробка])

Розглядаючи регіональний розподіл IT-витрат, також можемо побачити, що основними ринками є Північна Америка, Азія та Європа, в яких 78% IT-витрат зосереджено в цих регіонах – рис. 1.3 [60].

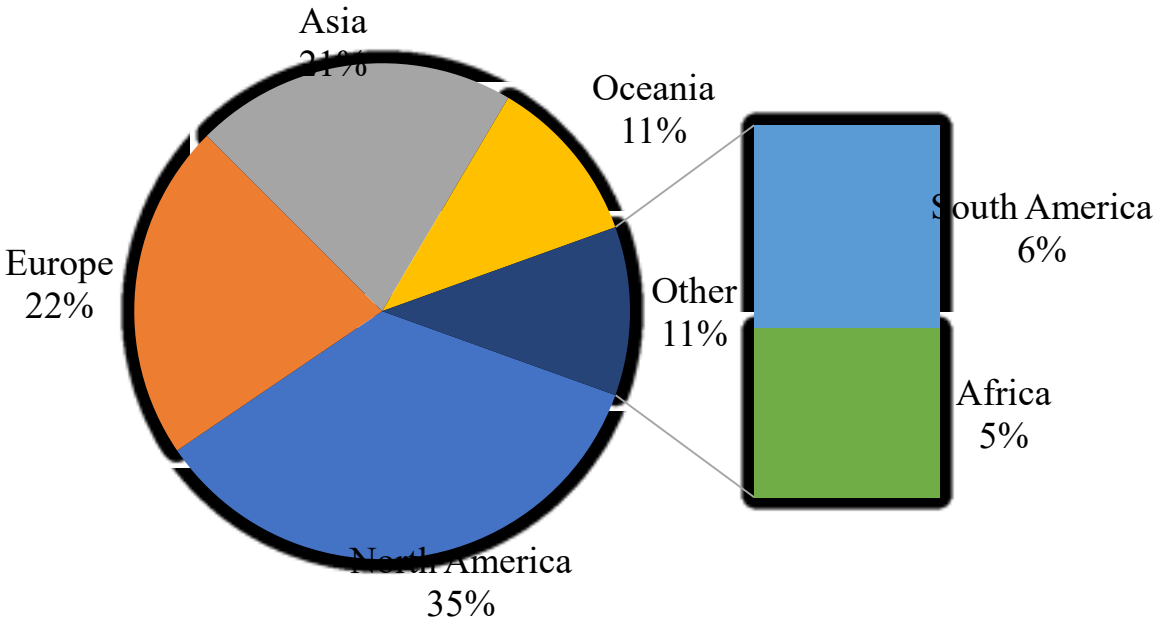


Рисунок 1.3 – Регіональний розподіл IT-витрат

(Джерело – [власна розробка])

Детальніше аналізуючи конкретні ринки, що зростають, можемо побачити, що найбільше зростання на 40% припадає на ринок корпоративного програмного забезпечення – табл. 1.1 [52].

Незважаючи на збільшення доступу до ІКТ з часом, частка доданої вартості цього сектору у світовому ВВП залишалася стабільною протягом останнього десятиліття, складаючи в середньому близько 4,5 відсотка [111]. Середнє світове значення може приховувати різні події в країнах, або що зниження цін на продукцію ІКТ супроводжувалось збільшенням обсягів. IT-витрати в розрізі світових ринків наведені у табл. 1.2.

Таблиця 1.2 – Світові IT-витрати за ринками за 2019-2021 рр (мільярди доларів США) (Джерело – [власна розробка])

IT-витрати	2019 Spending ($)	2019 Growth (%)	2020 Spending ($)	2020 Growth (%)	2021 Spending ($)	2021 Growth ($)
1	2	3	4	5	6	7
Data Center Systems	210	0.6	179	-14	191	6
Enterprise Software	476	12	529	11	735	40
Devices	711	-0.3	697	-1.9	808	16
IT Services	1040	5	1071	2.9	1207	12
Communications Services	1372	-0.6	1396	2	1458	4

На індивідуальному економічному рівні деякі економіки демонстрували зростання частки доданої вартості сектору ІКТ у своєму ВВП у період між 2010 та 2017 роками – рис. 1.4 [111].

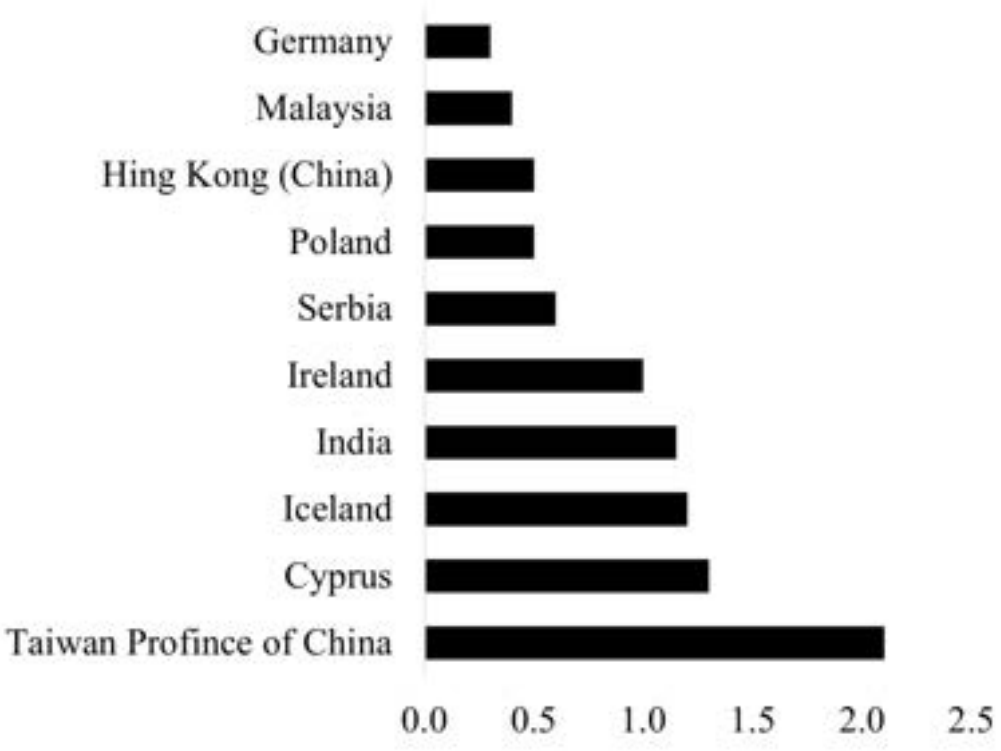


Рисунок 1.4 – Топ 10 зростання частки доданої вартості сектору ІКТ у ВВП за 2010–2017 рр (%)(Джерело – [111, с. 52])

Китайська провінція Тайвань посіла перше місце завдяки зростанню виробництва ІКТ. Індія посіла четверте місце, де зростання було зумовлене переважно комп'ютерними послугами.

Що стосується доданої вартості, Сполучені Штати мають найбільший у світі сектор ІКТ, майже вдвічі більший за другий за величиною Китай. Інші азіатські економіки, що входять у топ-10 за доданою вартістю, включають Японію, Республіку Корея, Індію та провінцію Тайвань – рис. 1.5 [111].

У секторі ІКТ комп'ютерні послуги є найбільшою складовою, із часткою приблизно 40 відсотків у загальній доданій вартості. Додана вартість у світовій індустрії комп’ютерних послуг є найбільшою в США – рис. 1.6 [111] – майже настільки великою, як загальна сума наступних дев’яти економік. Це пов’язано з тим, що сім найбільших у світі ІТ-компаній базуються в США, багато з яких отримують весь або більшу частину своїх доходів від комп’ютерного програмного забезпечення та послуг (наприклад, Microsoft, IBM та Oracle) [67].

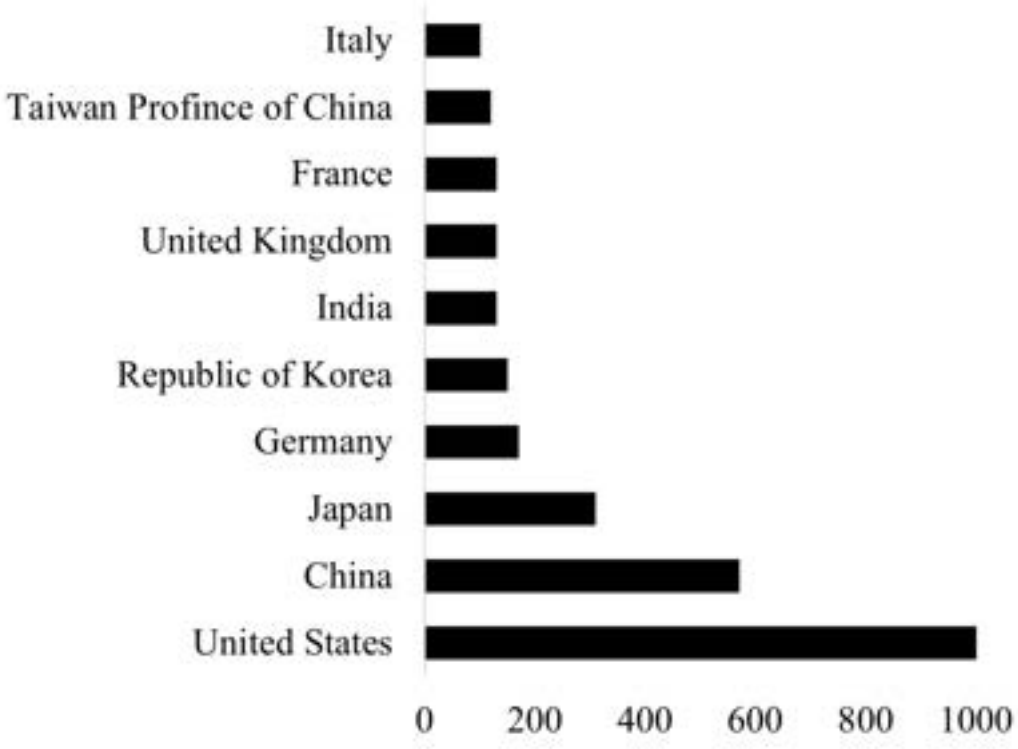


Рисунок 1.5 – Топ 10 доданих вартостей у секторі ІКТ за 2017 рік (мільярди доларів США) (Джерело – [111, с. 52])

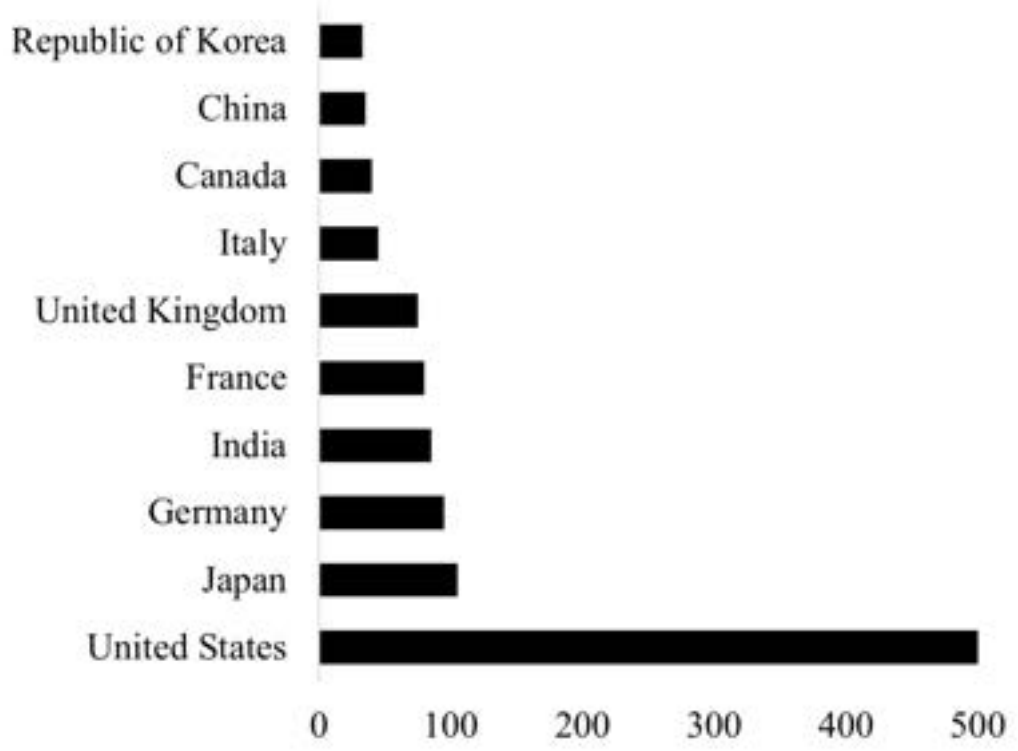


Рисунок 1.6 – Топ 10 доданих вартостей у комп'ютерних послугах за 2017 рік (мільярди доларів США) (Джерело – [111, с. 57])

Ірландія очолює список країн з точки зору доданої вартості комп'ютерних послуг, як частки від ВВП, із досить значним відривом, що відображає сильну присутність регіональних представництв цифрових компаній США – рис. 1.7 [111]. Лише одна країна, що розвивається (Індія), присутня серед 10 найкращих. Окрім Індії та Ізраїлю, всі інші економіки, що мають високі рейтинги за цим показником, є європейськими.

Світова зайнятість у секторі ІКТ зросла на 16 відсотків між 2010 і 2015 роками, що становить зростання з 34 мільйонів до 39,3 мільйона працівників. Як результат, частка ІКТ сектору в загальній зайнятості зросла з 1,8% до 2%. Зайнятість у комп'ютерних послугах зростала особливо швидко за той же період, на 27 відсотків, і становила найбільшу частку зайнятості в секторі ІКТ у 2015 році – рис. 1.8 [111]. Глобальна зайнятість у секторі ІКТ зросла на 37% за останні п'ять років та досягла більше 53 мільйонів осіб у 2020 році [80].

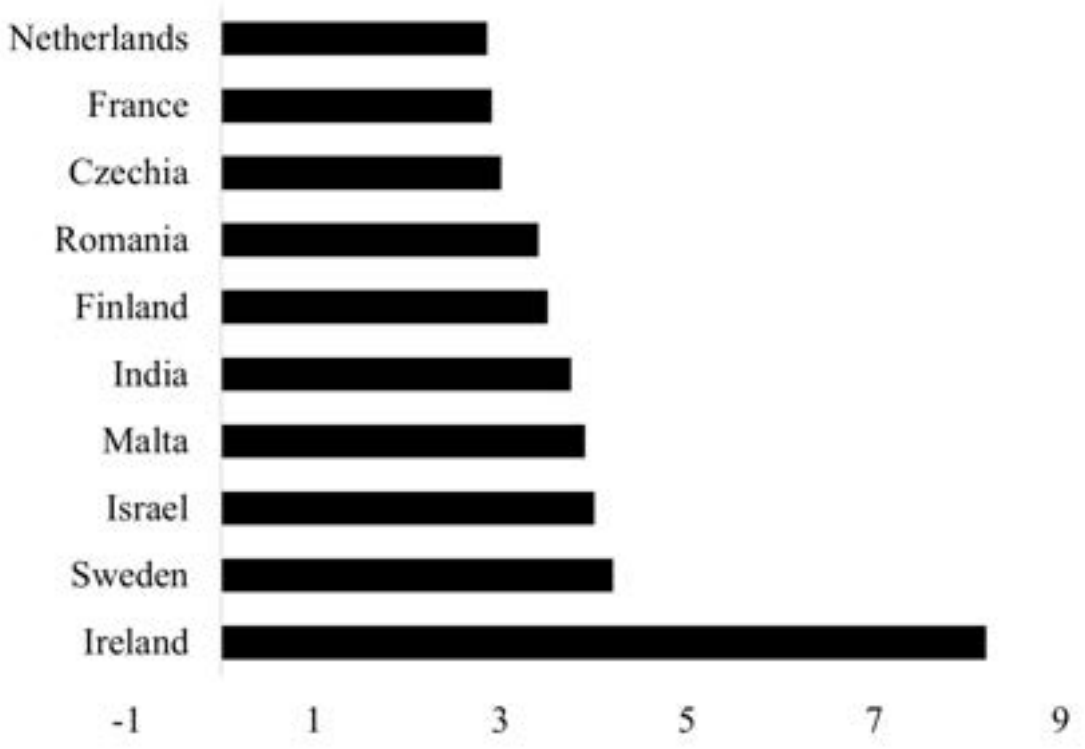


Рисунок 1.7 – Топ 10 доданих вартостей у комп'ютерних послугах як частка ВВП за 2017 рік (%) (Джерело – [111, с. 57])

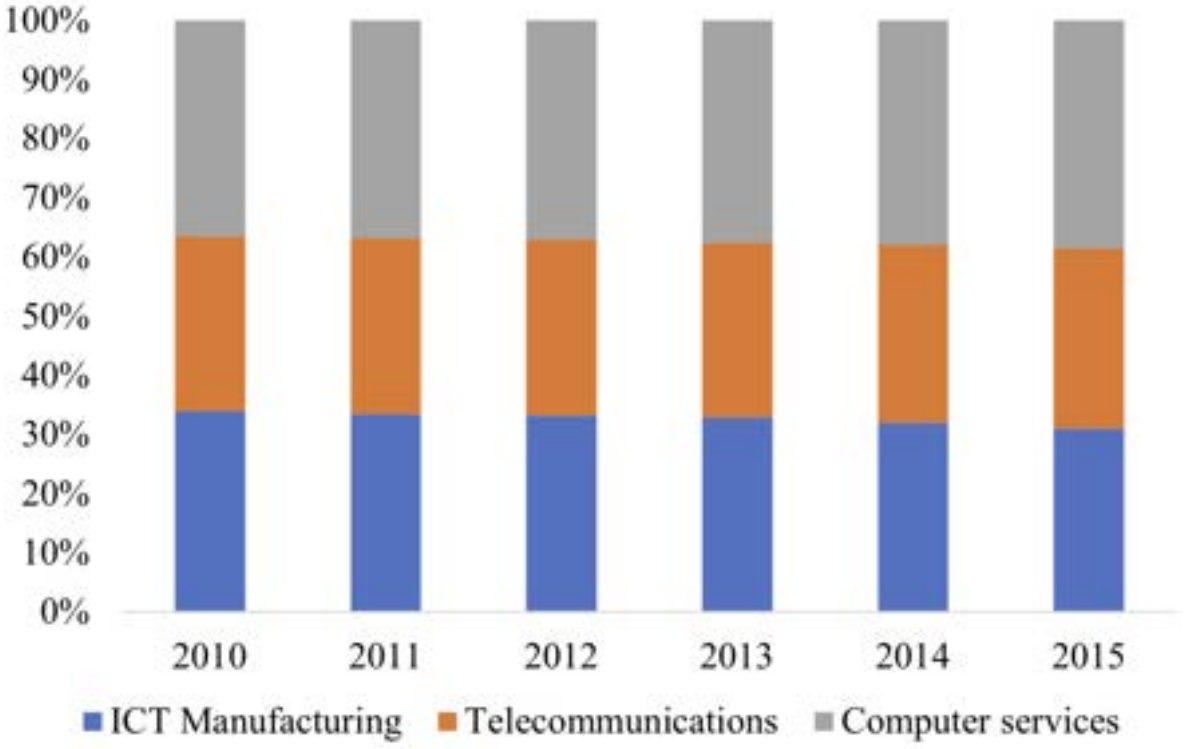


Рисунок 1.8 – Розподіл зайнятості у світовому секторі ІКТ за підгалузями за 2010–2015 рр (%) (Джерело – [111, с. 59])

ІКТ став найбільш динамічним сектором серед комерційних послуг у 2016 р. У період 2013–2019 рр. середньорічний темп приросту експортованої вартості ІКТ послуг склав 10% – рис. 1.9 [121, 122].

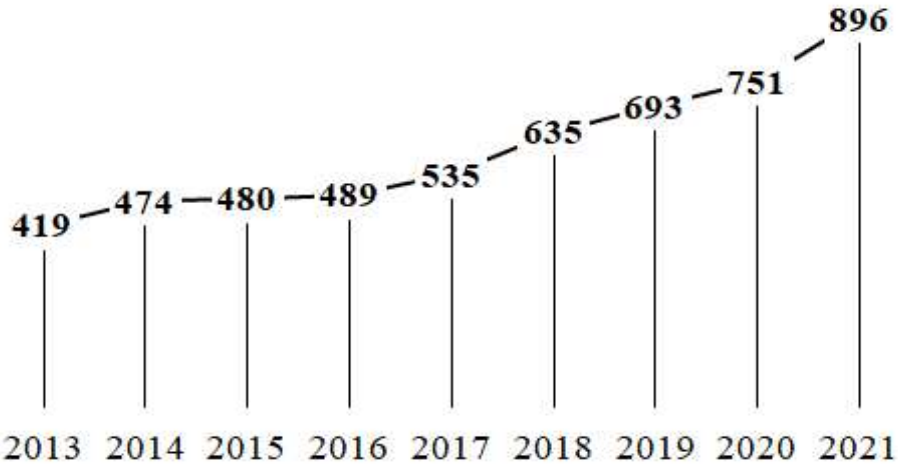


Рисунок 1.9 – Світовий експорт телекомунікаційних, комп'ютерних та інформаційних послуг за регіонами за 2013–2021 рр (мільярди доларів США) (Джерело – [власна розробка])

За оцінками Секретаріату СОТ (Світової організації торгівлі), в 2021 році комп'ютерні послуги (що складаються з апаратних і програмних послуг та послуг з обробки даних) становили близько 81 відсотків експорту ІКТ-послуг, 730 млрд. дол. Послуги телекомунікацій (включаючи мобільні телекомунікації, магістральні послуги Інтернету та надання доступу до Інтернету) оцінювались у 14 відсотків експорту ІКТ. Інформаційні послуги (включаючи послуги баз даних, такі як концепція, зберігання та розповсюдження інформації та послуги інформаційних агентств) становили лише 5 відсотків загального експорту ІКТ – рис. 1.10 [121, 122].

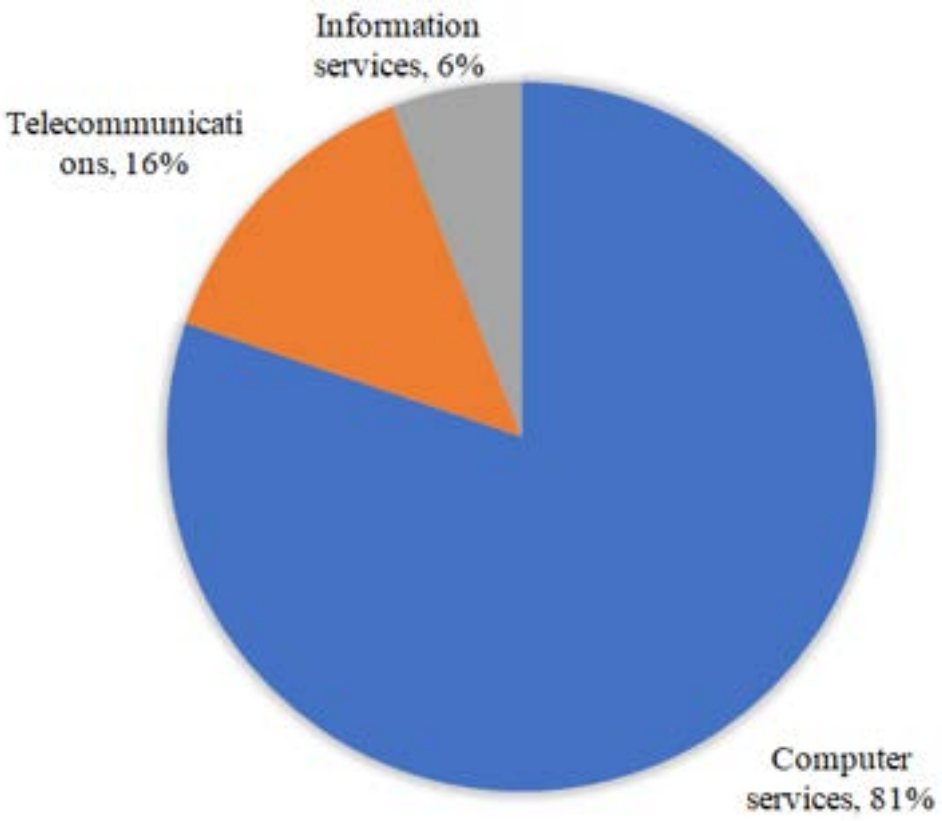


Рисунок 1.10 – Світовий експорт ІКТ за видами за 2021 рік
(Джерело – [власна розробка])

Деякі країни успішно розвивають експортну галузь у сфері послуг ІКТ. Серед провідних експортерів відносна важливість послуг ІКТ у загальному обсязі експорту послуг значно варіюється – рис. 1.11 [114].

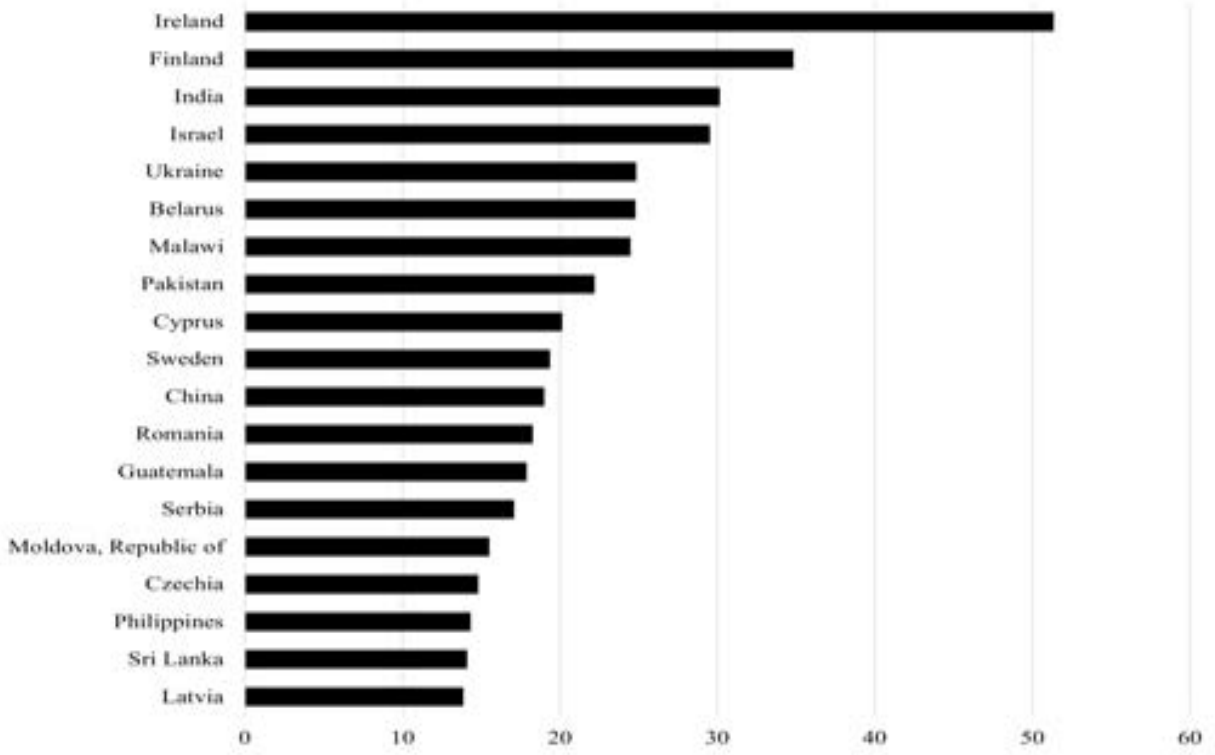


Рисунок 1.11 – Частка послуг ІКТ у загальному обсязі експорту послуг: 20 найкращих країн за 2019 рік (%) (Джерело – [власна розробка])

У Фінляндії, Індії, Ірландії та Ізраїлі ця частка перевищує 25 відсотків. Майже весь експорт послуг ІКТ в Ірландію здійснюється за допомогою комп'ютерних послуг. Частково це є статистичною аномалією, яка відображає присутність декількох транснаціональних цифрових компаній, що звітують про результати бізнесу в цій країні [63, 103]. В Індії, що посідає третє місце, майже третину з експортованих послуг становлять комп'ютерні послуги.

Пандемія COVID-19 домінувала над світовим економічним розвитком протягом 2020 та 2021 роками. Обмеження руху та інші заходи з охорони здоров’я населення зменшили глобальну економічну активність, впливаючи на виробництво, розподіл та споживання. Світова економіка за 2020 рік скоротилася на 4,3% – у два з половиною рази більше, ніж під час світової фінансової кризи 2008–2009 років [120]. Як і у випадку з усією економікою, пандемія мала майже негайний вплив на фінансові показники діяльності компаній в галузі ІТ. Загальні витрати на ІТ збільшились у 2020 році на 1,4% замість прогнозованого збільшення на 3,4% в порівнянні з 2019 роком. З іншого

боку пандемія сприяла переходу до віддаленої роботи та підвищеної цифровізації, і це в свою чергу привело до значного економічного росту IT індустрії у 2021 році.

Оскільки наслідки COVID-19 продовжують тиснути на глобальну економіку та посилює невизначеність на ринку, галузь управління проєктами відіграє цінну роль у забезпеченні досягнення проєктами своїх стратегічних цілей підтримки економічного зростання, створення робочих місць та стійкої інфраструктури. Незважаючи на все, що змінилося в 2020-2021 роках, одна річ залишилася незмінною – важливість технологій для успіху бізнесу. Пандемія COVID-19 продемонструвала значення ІКТ для забезпечення економічної діяльності в період різкого скорочення руху товарів і людей. Страх перед вірусом спонукав населення захиститися вдома та зменшити пряму, особисту взаємодію з іншими. Цифрові технології дали людям можливість проводити багато звичайних заходів віддалено, таких як робота, навчання, покупки та отримання медичних послуг.

IT-індустрія є відносно молодою для України галуззю, проте вона швидко стала важливим елементом у створенні іміджу країни. За останні роки IT-компанії України, в тому числі аутсорсингові компанії, опинилися на перших позиціях у багатьох впливових міжнародних рейтингах [42, 49, 55, 104-106, 109, 123].

Статистика, представлена різними організаціями, такими як Світовий банк (World Bank), PwC, Асоціація IT Україна (IT Ukraine Association), Державна служба статистики України (State Statistics Service of Ukraine) та НБУ демонструє стрімке зростання IT-галузі України. Інформаційно-комунікаційні технології (ІКТ) під час пандемії вперше стали провідною за величиною галуззю експортних послуг. Експорт ІКТ зріс на понад 36% за 2021 рік та досяг понад 8% всього українського експорту [7, 115]. У 2019-му році 60% українських розробників були працівниками аутсорсингових IT-компаній створення програмного забезпечення, 29% займалися розробкою продуктів в рамках продуктових компаній та 11% аутстаффінгом [110].

Український аутсорс IT-ринок демонструє стабільне зростання. Згідно з аналізом компанії PwC за період з 2011 по 2015 роки український ринок IT-послуг зріс у 2,5 рази (до 2.7 млрд. доларів та більш ніж 90 тис. спеціалістів) [27]. На 2021 рік він перевищав 6 млрд. доларів та 280 тис. Професіоналів [61].

Україна стає провідним IT-аутсорсинговим напрямком в Європі завдяки бурхливій екосистемі технологій, великому фонду талантів та сильній освіті [99]. Українські розробники програмного забезпечення визнані одними з найсильніших IT-професіоналів у всьому світі завдяки їх міцній технічній експертизі, високому рівню володіння англійською мовою та знанням нових дисциплін, таких як AI, big data, blockchain, тощо. Розглянемо детальніше, чому саме всесвітньо визнані бренди наймають розробників в Україні порівняно з іншими країнами Центральної та Східної Європі.

Глобальне визнання у технологічній галузі. Україна здобула переконливі позиції у кількох міжнародних технічних рейтингах. Країна займає 20-ту позицію в глобальному індексі розміщення сервісів (Global Services Location Index by A.T. Kearney) [42]. У 2017 році Глобальна асоціація пошуку ресурсів (Global Sourcing Association) Великобританії назвала Україну Великобританським «Місцем офшорного призначення року» [109]. Більше того, Міжнародна асоціація аутсорсингових спеціалістів (International Association of Outsourcing Professionals) у 2020 році занесла 21 українських IT-компаній до списку «Глобальний аутсорсинг 100» [123].

Технологічна освіта. Асоціація «IT Ukraine» повідомляє, що щорічно в Україні понад 20 000 студентів закінчують місцеві університети зі спеціальностями в галузі IT [61].

Кількість та якість. Українське співтовариство з інформаційних технологій та програмного забезпечення складається з більш ніж 280 000 професіоналів [61]. Але справа не лише в кількості. SkillValue визнав українських програмних інженерів одними з найкращих у світі із середнім показником оцінок 91,26 відсотків [119].

Дім всесвітньо відомих стартапів. PetCube, Gitlab, YouTeam, AXDRAFT, People.ai та Earth.ai – це українські стартапи потрапити в Y Combinator.

Таким чином, на початок 2020-их років ринок українського IT-аутсорсингу знаходився у стані активного розвитку, але Російсько-Українська війна внесла свої корективи. Дуже цікавим є питання щодо можливого майбутнього впливу цієї війни на IT-індустрію України та на економіку країни в цілому. На тлі кризи COVID-19, яка сильно вплинула на ринки праці в усьому світі, ця війна стала джерелом нового світового фінансового шоку для світу. Так як ситуація є багатофакторною, як політично так і глобально-економічно, робити достовірні припущення досить важко. Враховуючи переважаючу мінливість, подальші прогнози зосереджені на впливі війни на доходи пересічних громадян України, а саме на впливі скорочення ВВП України на ці доходи. UNDP було розглянуто 4 основні сценарії [41, 45]:

1. Скорочення на 7%. Базується на збройних конфліктах у 45 країнах Африки на південь від Сахари протягом 1989-2017.
2. Скорочення на 15%. Базується на скороченні ВВП, яке спостерігалося в Україні під час початкової фази військового конфлікту на південному сході України в 2014-15 роках.
3. Скорочення на 20%. Базується на скороченні ВВП, яке спостерігалося у війнах у Югославії чи Грузії, та на економічних доказах широкомасштабних воєн.
4. Скорочення на 60%. Це крайнє припущення, яке ґрунтується на попередньому досвіді Oxford Economics. У новітній історії, наприклад, економіка Лівану скоротилася приблизно на 40% у 1982 (війна в Лівані) і 1989 (війна за визволення); більш ніж на 50% у Південному Судані в 2012 році (криза Хегліга); і в Лівії на 53% у 2014 році (друга громадянська війна), на 60% у 2020 році (друга громадянська війна) і на 67% у 2011 році (військове втручання).

Ранні прогнози ПРООН (UNDP) свідчать про те, що в середньостроковій перспективі регресії в розвитку України будуть значними. Бідність населення

зростатиме та постраждає економіка країни і її соціальна структур. За оцінками уряду України, було знищено інфраструктуру, будівлі, дороги, мости, лікарні, школи та інші фізичні активи на суму щонайменше 100 мільярдів доларів. Війна призвела до повної зупинки 50% українських підприємств. У разі затягування війни до 90% населення України може опинитися за межею бідності. Симуляція прогнозує наслідки зменшення доходів домогосподарств не менше 15% (сценарій 2). За цим сценарієм через 12 місяців очікується, що понад 6% населення житимуть за межею бідності в 5,50 доларів США на день. 54% населення житиме з високим ризиком повернутися до бідності, тобто під межею вразливості в 13 доларів США на день. У разі продовження та поглиблення тривалої кризи прогнози свідчать про те, що 90% населення зіткнеться з бідністю або вразливістю до бідності (сценарій 4). Цей прогноз означає втрату 18 років соціально-економічних досягнень в Україні та повернення до рівня бідності, який востаннє спостерігався у 2004 році.

Неминучим є зниження рівня життя, скорочена доступність базових речей та послуг та підвищення цін та загальна регресія розвитку України. Чим тривалішою буде війна, тим більшим буде рівень регресії. Таким чином, ІТ-фахівці, які були вимушені виїхати за кордоном та все ще працюють на українські ІТ-компанії, можуть вирішити розірвати контракти з українськими компаніями та емігрувати на постійній основі, що завдасть шкоди ІТ-сектору. Потенційне скорочення українського ІТ-сектору негативно вплине на економіку України та знизить спроможність до економічного відновлення країни, але це далеко не єдиний фактор, який матиме аналогічний ефект. Не варто забувати, що майже 2 мільйони працездатних біженців мають вищу освіту та можуть прийти до аналогічних висновків щодо еміграції на постійній основі [58]. В комбінації з тим, що 2 мільйони біженців – діти та підлітки, це створює потенціал для відтоку як теперішнього так і потенційно майбутнього інтелектуального потенціалу країни, в тому числі потенційно майбутніх ІТ-фахівців. Значним є вплив Російсько-Української війни на світову економіку. Високе навантаження на ринки праці сусідніх з учасниками війни країн,

зростанням дефіциту робочої сили в Європі в пост-ковідному просторі, рекордне зростання світового індексу цін з моменту початку війни, створило умови для пошуку більш безпечних ІТ-ринків для інвестицій, наприклад Індія та Центральна Азія, та переманювання високо-кваліфікованих українських ІТ-спеціалістів.

Ситуація, що склалася, призводить до необхідності аналізу процесу управління аутсорсинговими ІТ-проєктами з метою підвищення його ефективності на всіх стадіях. Водночас, зростання кількості проєктів створює для ІТ-компаній значні труднощі, які згідно зі специфікою ІТ-сфери, значно відрізняються від загальних проблем в області управління проєктами. Інструменти управління, необхідні для керування портфоліо проєктів, пропонуються багатьма методиками. Однак, незважаючи на паралелі між програмним забезпеченням та іншими галузями інженерії, відмінні риси ІТ проєктів роблять стандартні методи управління недоречними. Все це призводить до необхідності розробки й впровадження нестандартних підходів, які б поєднали сучасні технології, існуючі в ІТ-галузі, з методами врахування різноманітних змін у середовищі розгортання як окремих проєктів, так і комплексних портфоліо. Одним з напрямків удосконалення процесів управління може стати залучення методів нечіткої математики. З огляду на цю інформацію є доречним та своєчасним більш детально розглянути процес управління проєктами аутсорсингових ІТ-компаній.

1.2 Ключові аспекти та класифікація проблем управління проєктами ІТ-компаній

На сьогоднішній день існує велика кількість ІТ-компаній, що відповідають за розробку великої кількості проєктів одночасно. Для цього використовуються сучасні технології та методи, націлені на спрощення та

покращення управлінської діяльності. Однак величезна кількість методик управління лише змінює спосіб прийняття рішень, а не спрощує щоденну рутину їх прийняття. Хоча в рамках управління проєктами було зібрано величезний обсяг знань та розроблено певну кількість методик, в межах яких пропонуються численні інструменти управління, необхідність керувати зростаючою кількістю різноманітних проєктів створює для ІТ-компаній все більше труднощів. Згідно зі специфікою діяльності ІТ-фірм ці труднощі різко відрізняються від загальних проблем в області управління проєктами.

Менеджмент проєкту часто називають “агентами змін”, тому що вони змушені контролювати протікання проєкту в умовах постійних змін. Саме тому дослідники процесу управління проєктами, в першу чергу, намагаються виділити загальні процеси, присутні у кожному проєкті. Якщо розглянути процес управління проєктами, то можна прийти до висновку, що усі проєкти об’єднує факт виникнення та подолання проблем в процесі управління проєктами. Рис. 1.12 демонструє ключові аспекти управління проєктами, що дозволять створити класифікацію проблем управління проєктами. На основі цієї класифікації виникає можливість проаналізувати доцільність використання того чи іншого математичного апарату для подолання конкретних проблем.

Обмеження, накладені на окремі проєкти, є основою для подальшої класифікації проблем, притаманних процесу управління портфелем проєктів. На даний момент у більшості методик виділяють наступні “потрійні обмеження” [90]: визначення продуктивності, графік роботи та бюджет. Ситуація стає більш складною зі збільшенням кількості проєктів, тому перед узагальненням класифікації проблем управління портфелем проєктів є сенс розглянути проблеми, з якими стикаються менеджери окремих проєктів.

ІТ-проєкти не підпадають під дію обмежень так само, як проєкти, результатом яких є фізична продукція, наприклад, проєкти будівництва. І все таки програмне забезпечення регулюється реальними обмеженнями, але вони, як правило, мають багатовимірний та абстрактний характер, тому буває досить важко їх зрозуміти та обговорювати. Проте як замовник, так і виконавець часто

забувають або просто не розуміють обмежень ІТ, що призводить до нереалістичних очікувань та надмірно амбітних проєктів [86]. Це пов'язано, принаймні частково, із враженням, що програмне забезпечення в значній мірі не містить обмежень, і тому його потенціал є необмеженим. Далі проведемо аналіз обмежень, що найчастіше зустрічаються при управлінні ІТ-проєктами.

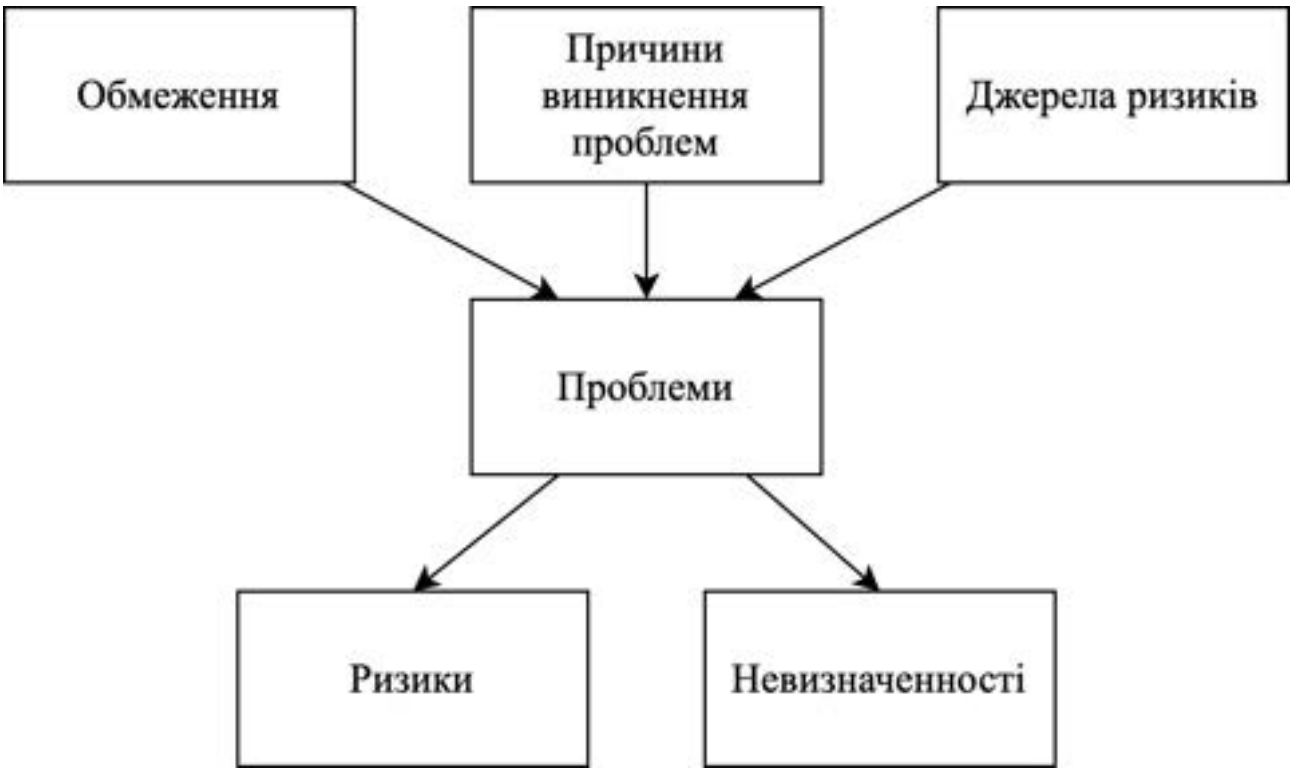


Рисунок 1.12 – Ключові аспекти класифікації проблем управління проєктами (Джерело – власна розробка)

Візуалізація. Програмне забезпечення фактично невидиме. Проблема візуалізації є джерелом багатьох потенційних недоліків ІТ-проєктів. Також надзвичайно важко представляти ключові аспекти програмного забезпечення доступним для всіх стейкхолдерів чином, що робить процес специфікації потенційно складним. У випадку будівлі досить легко створити фізичне подання, яке може обговорюватися стейкхолдерами та використовуватися як схема. Для специфікації програмного забезпечення існує багато графічних програмних додатків, наприклад уніфікована мова моделювання (Unified Modelling Language, UML), однак вони не однозначні і описують обмежені

аспекти системи. Ще одна складність, пов'язана з "невидимістю" програмного забезпечення, зустрічається під час моніторингу проєкту. Відсутність матеріального продукту означає, що проєкт може тривати протягом значного часу, перш ніж проблеми стануть очевидними, без можливості перевірити, що витрати часу та грошей співвідносяться з прогресом проєкту в бажаному напрямку.

Гнучкість. Ще однією проблемою IT-проєктів, що також пов’язана з нематеріальним характером програмного забезпечення, є зловживання сприйнятою гнучкістю програмного забезпечення. Неможливість візуалізувати межі того, що можливо або практично в IT, заохочує замовників змінювати своє уявлення стосовно проєкту частіше, ніж в інших інженерних проєктах, де обмеження більш очевидні. Надмірні запити щодо нових функцій або зміна функцій під час виконання проєкту створюють непотрібну та небажану складність. Це сприяє перевитраті часу та бюджету, тим самим збільшуючи ймовірність провалу проєкту. Гнучкість також відображається в тому, що в інженерії програмного забезпечення, як правило, існує кілька способів вирішення однієї і тієї ж проблеми, на відміну від, наприклад, будівництва, де, як правило, визначаються більш чітко встановлені процеси. Це призводить до частої зміни уявлення стосовно проєкту зі сторони виконавця, що також в свою чергу сприяє перевитраті часу та бюджету.

Складність. Складність та заплутаність може стати значною перешкодою для успішного проєктування та реалізації IT-проєктів. Хоча у великих проєктах інших інженерних дисциплін, очевидно, також доводиться стикатися зі складністю, здається, що в галузі розробки програмного забезпечення її водночас і важче виявити, і важче зрозуміти. Частина складності проєкту є обґрунтованою та явною, тобто необхідною для виконання вимог. Але є неочевидні складності стосовно реалізації проєктів, які значно складніше виявити, адже вони можуть бути унікальними в залежності від предметної області, типу проєкту та технологій, які використовують виконавці. Ступінь складності, пов'язаної з досягненням певної мети, може бути дуже важко

оцінити на початку проєкту. Як результат, проєкти можуть стати набагато складнішими, ніж було спочатку заплановано; такі проєкти надзвичайно чутливі до невдач або труднощів.

Невизначеність. Багато складних ІТ-систем прагнуть виконувати або спрощувати завдання, які раніше виконували люди. З'ясування чітких вимог до таких систем може викликати великі труднощі. Вже після началу проєкту може виявитися, що розробка триватиме довше або коштуватиме набагато більше, або є певні функції, які неможливо доставити.

Незважаючи на незаперечні паралелі між програмним забезпеченням та іншими галузями інженерії, існує аргумент, що відмінні риси ІТ роблять стандартні методи управління недоречними. Наприклад, вже давно встановлено, що збільшення розміру команди вище за певну кількість, ймовірно, зменшить ефективність розробки програмного забезпечення, через складність обміну інформацією між членами команди у ситуаціях, коли їх робота дуже взаємозалежна. Крім того, оскільки дуже важко точно визначити необхідні специфічні особливості складної ІТ-системи на початку проєкту, необхідно застосовувати ітеративні підходи набагато частіше, ніж у звичайних інженерних проєктах. Наприклад, ІТ-проєкти, як правило, передбачають постійний зворотний зв'язок між етапами проєктування та реалізації.

Тепер можемо розглянути еволюцію проблем управління ІТ-проєктами зі зростанням їх кількості, або, якщо бути точніше, проблеми управління портфелем ІТ-проєктів. Управління портфелем проєктів – комплексне поняття, яке включає в себе ряд ключових проблем, вирішення яких забезпечується технологіями управління портфелем. Процес управління портфелем проєктів дозволяє організаціям одночасно виконувати декілька пов'язаних проєктів і отримувати значні вигоди. Група проєктів, які не мають загальної точки керування, швидше за все не можуть досягти бажаного результату. Традиційні моделі і методи управління проєктами не визнають реальність сьогоднішніх компаній та некоректно тлумачать пріоритети на робочому місці, а також не впливають на потенційні переваги. Процес управління портфелем проєктів

урівноважує основні обмеження портфелю та забезпечує інструмент для прийняття рішень протягом усього циклу виконання портфелю на основі показників ефективності, встановлених процедур та цілей портфелю. Поліпшення процесу управління сприяє зменшенню експлуатаційних витрат та підвищенню рентабельності усього портфелю проєктів та зокрема окремих проєктів. Тому управлінський персонал на рівні портфелю проєктів може спостерігати та керувати за повним обсягом роботи та діапазоном дій, необхідних для ведення різної кількості проєктів, дотримуючись бюджету, розкладу та специфікацій кожного з проєктів.

Головною роллю управлінського персоналу портфелю проєктів є нагляд за багатьма менеджерами проєктів. Цей персонал, як правило, несе відповідальність за попереднє проєктування, проєктні бюджети та розклади, адміністрування договорів, нагляд за процесом виконання портфелю та окремих проєктів. Отже менеджер портфелю має більший спектр обов'язків у порівнянні з менеджером проєкту. Він працює на більш високому рівні та керує багатьма менеджерами проєктів одночасно. Для того, щоб знайти кращі рішення та спростити процес виконання портфелю, включаючи планування, розподіл ресурсів, таких як персонал та матеріальні ресурси та мінімізація витрат при досягненні цілей портфелю, необхідно визначити найбільш повний та виключний список проблем, з якими може зіткнутися управлінський персонал у процесі виконання портфелю проєктів.

При управлінні портфелем проєктів необхідно збалансувати часто суперечливі вимоги з обмеженими ресурсами та координувати націленість з метою забезпечення оптимального загального результату, або іншими словами необхідно фокусуватись на глобальному оптимумі.

Таким чином, питання управління кількома проєктами створює новий набір проблем, котрі компанії необхідно вирішити.

Однією з перших проблем, з якою стикаються проєктні менеджери, є проблема нестачі ресурсів. Від доступності ресурсів будуть залежати ключові характеристики проєкту: тривалість, вартість і якість. В даному випадку

ресурси можуть бути трьох типів: трудові, матеріальні, фінансові. Технологія, яка забезпечує проєкти ресурсами, називається розподіл (призначення або вирівнювання) ресурсів. Точніше, існують дві технології призначення ресурсів: одна використовується на етапі формування програми або портфеля, інша – на етапі виконання (оперативне планування). Остання технологія також називається усуненням конфліктів між ресурсами. Брак ресурсів може змусити призупинити або навіть припинити виконання проєкту. Подібні задачі виникають тоді, коли проєкти портфеля спільно використовують загальні ресурси. Тобто саме використання загальних ресурсів може змусити перейти від моделі управління окремими проєктами до моделі управління портфелем проєктів. Визначення проблем та взаємозв'язків між ними в компанії складає основу для їх подолання, і одночасно з цим, виставляє на передній план області досліджень управлінських рішень.

Попередньо було розглянуто обмеження, які виникають в процесі управління ІТ-проєктами – перша складова, яка дозволяє сформувати набір проблем управління ІТ-проєктами. Для повного розуміння цих проблем також необхідно розглянути другу складову їх формування – причини їх виникнення. Виділяються наступні причини виникнення проблем в процесі управління комплексами проєктів [90, 107]:

- Неналежний рівень підготовки до проєкту: труднощі планування детального та реалістичного процесу розробки проєкту;
- Несвоєчасне спостереження за прогресом проєкту: цілі проєкту змінюються впродовж його виконання; упущення моменту запиту на нестачу або надлишок ресурсів;
- Вибір занадто великих проєктів: проєкт не розділено на менші проєкти;
- Дублювання задач: рамки проєкту розмиті або невизначені; задачі дублюються між проєктами;
- Несвоєчасний збір інформації: упущення моменту збору інформації з проєктів; відсутність інформації з проєктів;

- Прийняття некомпетентних рішень: прийняття рішень без належної оцінки наявних ресурсів та пріоритетів проєктів;
- Нагромадження проєктів;
- Проблеми, пов'язані з ролями та обов'язками;
- Проблеми, пов'язані з ресурсами.

Як бачимо, перерахований обсяг причин виникнення труднощів при управлінні портфелю проєктів є досить великим, але при цьому і не повністю вичерпним. Замість того, щоб поповнювати список ймовірних або типових причин труднощів має сенс перейти до розгляду результуючих проблем, як результат комбінування обмежень проєктів та причини виникнення проблем при управлінні проєктами. Маючи список проблем, з якими стикається менеджер портфоліо проєктів, можна створити їх більш вичерпну класифікацію.

Було проведено дослідження основних проблем пов'язаних з портфелем проєктів. Аналіз літератури свідчить про те, що є досить багато проблем, пов'язаних з управлінням портфелем проєктів. Можна виділити наступні проблеми у процесі управління портфелем проєктів [1, 30]:

- Занадто багато проєктів для наявних ресурсів;
- Відсутність персоналу різних рівнів та навичок.
- Робота над проєктом набуває другого плану через велику кількість “офіційних” повноважень.
- Перекриття завдань в проєктах.
- Цілі різних проєктів не враховані в загальній стратегії.
- Рідко відбувається перегляд зв’язків між проєктами.
- Відсутні пріоритети проєктів.
- Нестаток обміну інформацією між проєктами.
- Відсутність загальної бази знань або даних для проєктів.

Даний список ні в якому разі не можна вважати повним, але й подальше його нагромадження не має сенсу, адже для того, щоби мати більш комплексну точку зору на це питання краще розглянути питання класифікації цих проблем.

У літературі зустрічається досить багато варіацій до підходів класифікування цих проблем. Було виділено два підходи, які містять та охоплюють найбільш повний обсяг проблем управління. Перший підхід групує проблеми за проблемними областями [30, 34, 43]:

1. Ролі та обов'язки:
    a. Нечіткі ролі та обов'язки: невизначено керівника проєкту (портфелю); невизначено рамки ролей та обов'язки управлінського персоналу;
    b. Часті зміни у структурі управління проєктом (портфелем): проблемно визначити персонал, відповідальний за виконання конкретного завдання;
    c. Відсутність мотивації: неналежне виконання завдань управлінським персоналом;
    d. Відсутність контролю процесу управління: велика кількість неналежно розподілених завдань призводить до перевантаження управлінського персоналу
2. Ресурси:
    a. Недієві методи оцінки: вибір недієвих методів оцінки ресурсів, необхідних для виконання проєкту (портфелю);
    b. Підбір персоналу: проблеми підбору компетентного персоналу для виконання проєкту; проблеми відбору компетентного персоналу для управління проєктом (портфелем);
3. Інформація:
    a. Нестача інформації по проєктам;
    b. Невизначено шляхи обміну інформацією між проєктами у портфелі та періодичність обміну;
    c. Відсутність самонавчання: при формуванні нового проєкту (портфелю) відсутня інформація, стосовно старих проєктів (портфелів).

Аналізуючи сформовані проблемні області було виявлено, що досить багато проблем управління портфелем проєктів виникають з проблем управління окремих проєктів цього портфелю – рис. 1.13.

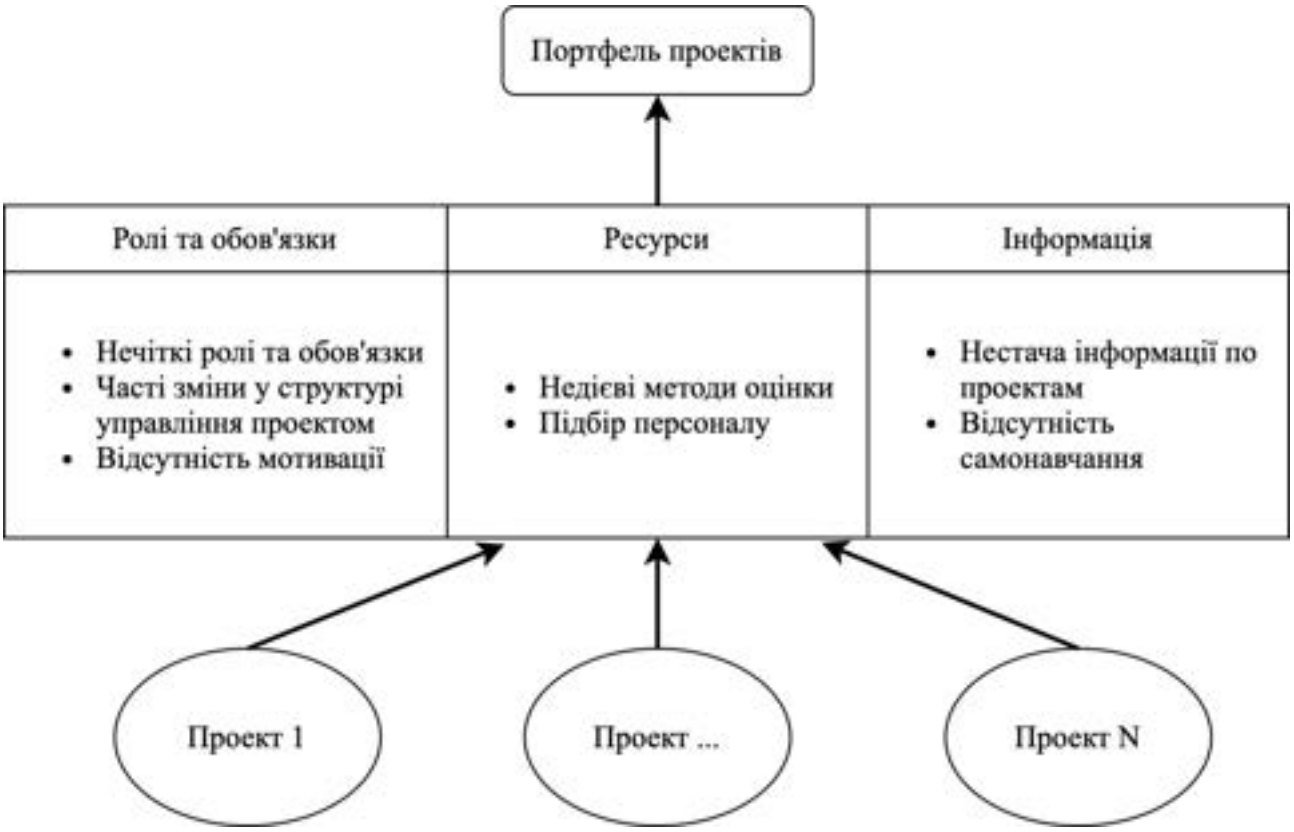


Рисунок 1.13 – Успадкування проблем управління портфелем з окремих проєктів (Джерело – власна розробка)

Другий підхід до класифікації проблем управління портфелем проєктів розділяє ці проблеми на управлінські рівні:

1. Проблеми діяльності на рівні розробки проєкту:
    a. Неналежний рівень підготовки до проєкту: труднощі планування детального та реалістичного процесу розробки проєкту;
    b. Несвоєчасне спостереження за прогресом проєкту: цілі проєкту змінюються впродовж його виконання; упущення моменту запиту на нестачу або надлишок ресурсів;
    c. Вибір занадто великих проєктів: проєкт не розділено на менші проєкти;
2. Проблеми діяльності на рівні портфелю проєктів:

a. Дублювання задач: рамки проєкту розмиті або невизначені; задачі дублюються між проєктами поточного портфелю проєктів; задачі дублюються між різними портфелями проєктів;
b. Несвоєчасний збір інформації: упущення моменту збору інформації з проєктів у портфелі; відсутність інформації з проєктів у портфелі;
c. Прийняття некомпетентних рішень: прийняття рішень без належної оцінки наявних ресурсів та пріоритетів проєктів у портфелі;
d. Нагромадження портфелю: відсутні рішення щодо закриття нерентабельних проєктів у портфелі;
e. Проблеми, пов'язані з ролями та обов'язками;
f. Проблеми, пов'язані з ресурсами

Але навіть у разі успішного встановлення шляхів подолання сформульованих проблем виникає велика кількість задач, щодо відбору проєктів для формування «правильного портфеля», відбору виконавця обраного проєкту, базуючись на попередньому досвіді, що в свою чергу веде до виникнення ще більшого кола задач, для яких менеджеру необхідно прийняти рішення.

Ці завдання самі по собі досить складні, тому, в першу чергу, крім економічної ефективності окремих проєктів, необхідно також враховувати відповідність проєктів обраній стратегії. Тому одним з неодмінних умов для впровадження управління портфелем проєктів є наявність в компанії стратегічного бачення: місії, цілей і стратегії.

Управління ризиками є важливою частиною процесу управління окремими проєктами та стає більш важливим у разі формування стратегічного бачення, тобто в процесі управління портфелем проєктів. У зв’язку з цим, далі проведемо аналіз ризиків управління ІТ-проєктами.

Виділяють наступну класифікацію ризиків проєкту [90]:

1. *Технічні ризики:* ризики, пов'язані з проєктуванням та експлуатацією продукту та/або виробничих процесів проєкту.

2. *Логістичні ризики:* ризики, пов'язані з економічними змінами, постачанням, закупівлями, інвентарем, технічним обслуговуванням та підтримкою.
3. *Програмні ризики:* ризики, пов'язані з отриманням та використанням програмних та проєктних ресурсів, таких як технічні експерти, спеціальні інструменти та бюджети на капітальні та проєктні витрати.
4. *Комерційні ризики:* ризики, що змінюють припущення та впливають на доходи, витрати, частку ринку, прибутковість, тощо.

Усі проблеми, які підпадають під цю класифікацію вносять свій внесок у проблемоформуванні портфоліо проєктів. При управлінні численними проєктами з'являються проблеми, які не можуть виникнути при управлінні окремими проєктами. Маючи класифікацію ризиків є можливість виділити найбільш впливові ризики.

Конкуренція. Перший виклик, з яким нині можна зіткнутися під час розробки аутсорс проєктів – це висока конкуренція на ринку. Зайве нагадувати, що IT-індустрія швидко розвивається. У 2022 році розмір ринку аутсорсингу інформаційних технологій склав $92,5 млрд за даними Statista [62]. Галузь нагадує стихійний масовий рух тим, як багато людей потрапляють в IT з невеликим досвідом або без будь-яких конкретних навичок, але з високим бюджетом для інвестування. Але на відміну від золотої лихоманки, IT дозволяє різноманітно мислити та виражатися. Наприклад, це не обмежує людину в певних робочих місцях. Ось чому, щоб впоратися з цим викликом, багато компаній наймають фахівців, які не володіють ідеальними IT-навичками, але мають досвід макро- та мікроекономіки, психології, статистики та інших галузей. Знання того, як кодувати та розробляти, недостатньо, щоб виділитися з натовпу численних постачальників програмних послуг.

Оцінка обсягу. Розглянемо детальніше одне з найпоширеніших завдань розробки програмного забезпечення: оцінка обсягу проєкту. Неадекватне або погане визначення обсягу негативно корелює з результатами проєкту [74]. Більшість схильні робити оптимістичні припущення під час оцінки часу,

ресурсів та зусиль, необхідних для виконання завдання. У розробці програмного забезпечення правильна та адекватна оцінка обсягу робіт є основною умовою успіху та вимагає використання інструментів та методів управління проєктами. Як правило, приблизні розрахунки, які клієнти отримують від розробників, досить далекі від фактично витрачених годин, що призводить до накладних витрат та незадоволеності клієнтів, які не довіряють своєму провайдеру і вважають, що вони складають години, щоб заробити більше грошей. Фактори, які впливають на будь-яку початкову / грубу оцінку проєкту, є такими:

- Наявність детальних технічних вимог;
- Тип проєкту (незалежно від того, наскільки схожі проєкти здаються, кожен з них унікальний і може потребувати різних технологій та інструментів для виконання);
- Очікувані в додатку функціональні можливості та їх надійність;
- Складність інтеграції бекенду, тощо.

Це завдання стосується роботи в команді та розумного планування. Якщо керівнику проєкту бракує лідерських якостей та необхідних навичок для розумного та ефективного управління, це перша помилка. Етап планування будь-якої програми розробки програмного забезпечення є вирішальним кроком для розуміння того, який стек технологій та які IT-експерти потребує менеджер проєкту. Цей етап розробки проєкту також показує, наскільки добре команда розуміє потреби та вимоги свого клієнта до кінцевого продукту. Таким чином, розумна оцінка обсягу проєкту вимагає питань, безлічі питань і чітких відповідей. Він також вимагає чіткого та об'єктивного бачення інтеграції продукту, коли він готовий до випуску. Якщо команда розробки не відповідає власній оцінці, це може мати такі наслідки для компанії:

- Пропущені віхи та затримка випуску.
- Дефіцит бюджету.
- Понаднормові роботи.
- Поганий моральний дух команди.

- Втрачений дохід (особливо у випадку проєктів з фіксованою ціною).
- Нещасні та не лояльні клієнти.
- Багато стресу.

Додаткові фактори:

- Затвердження клієнтом технічних вимог не гарантує, що документ містить реалістичний обсяг роботи. Реалістичний обсяг роботи буде зрозуміло після прийняття проєкту від клієнта.
- Будь-яке завдання має вузькі місця, які можна інтерпретувати кількома способами.
- Рівень продуктивності та ефективності розробників може бути різним у колективі, оскільки кожен має свій робочий темп і звички, тому робити неможливо чітких прогнозів щодо того чи іншого результату роботи працівника під час розробки проєкту просто неможливо.

Комунікація. Зазвичай розробники отримують напівструктурні бізнес-вимоги на першій зустрічі з планування. Клієнт має уявлення про велику картину того, що необхідно від програмного забезпечення з певними чіткими вимогами щодо важливих особливостей. Деякі навіть мають конкретну технологію, яку вони хочуть включити в свій стек. Проблема полягає в тому, що розробники зіткнуться з проблемами, не опитуючи представників усіх зацікавлених сторін проєкту під час виявлення. Більшість запитів на зміни походять від неповного початкового розуміння того, що необхідного функціоналу. Найкращою особою, яка роз'яснить, чого хоче конкретна категорія зацікавлених сторін, є хтось із зацікавлених сторін проєкту. Без цього кроку розробники можуть несвідомо витрачати час на прототип, який не має необхідних функцій або створює надмірну складність. Спілкування допомагає ловити подібні помилки перед тим, як рухатися вперед. Недоліки планування та відсутність комунікації є найважливішою причиною невдач у розробці користувацького програмного забезпечення, набагато більше, ніж технічні некомпетентності чи недосяжні вимоги.

Перевищення витрат. Впровадження проєкту з розробки програмного забезпечення на замовлення може бути ризикованим процесом. В середньому 66% програмних проєктів з бюджетом понад $15 млн перевищують кошти, 33% перевищують запланований графік, а ще 17% призводять до дефіциту прибутку від проєкту [34]. На жаль, було помічено, що більшість проєктів, що передбачають розробку програмного забезпечення, страждають від перевиконання бюджету та графіків, викликаних, серед інших причин, недостатніми початковими оцінками [35, 83].

Навіть порівняно з IT-проєктами в цілому, аутсорс проєкти несуть найвищий ризик перевитрати з точки зору витрат і планування [34]. Причинами цієї проблеми є погана оцінка вартості та графіку або зміни обсягу проєкту, а не збій управління проєктом. Через непорозуміння та неправильну комунікацію можуть легко спливати сюрпризи, з якими можна було розправитися раніше, якби вони були належним чином виявлені. Такі помилки можуть виникати між усіма учасниками проєкту програмного забезпечення. Навіть якщо потенційна проблема буде виявлена на початку, це не є гарантією того, що необхідне спілкування відбуватиметься.

Наступним складним фактором, який слід враховувати, є зміна характеру вимог, яка може розвиватися протягом усього проєкту розвитку. Якщо проєкт розтягується занадто довго, це лише збільшує ризик зміни вимог. Це означає, що через недостатню комунікацію чи тривалі проєкти, розробку може знадобитися багато разів корегувати. Оскільки більшість основних дослідників ризику, таких як Макфарлан, Аддісон та Карні, визначають вимоги, що ґрунтуються на вимогах, як найвищий ризик програмного проєкту [29], чітке забезпечення вимог та строгості строків протягом усього процесу є важливим для усунення впливу потенційного майбутнього. Окрім того необхідно враховувати не тільки вимоги та потреби клієнта, а також потреби основної аудиторії.

Незважаючи на те, що вивчення та врахування усіх можливих вимог користувачів є вирішальним при розробці програмного забезпечення, дуже

часто команді розробників не вистачає найважливіших знань про людей, для яких будують продукт. Поточні вимоги користувачів можуть ігноруватися, що автоматично робить всю концепцію проєкту неправильною.

Технологічний стек. Після того, як проєкт має хороший план, його затвердили та схвалили зацікавлені сторони, постає одна з найскладніших проблем розвитку програмного забезпечення, яка полягає у знаходженні достатнього набору технологій. Для компанії з розробки програмного забезпечення важливо мати фахівців з великим досвідом та багатою палітрою навичок.

Зазвичай менеджер проєктів контролює команду таких професіоналів, але у великих компаніях працівники з високо оціненими знаннями можуть бути залучені до роботи над кількома різними проєктами. Однак отримати достатній технічний стек для проєкту – це не лише завдання керівника проєкту.

Настійно рекомендується проконсультуватися з рекрутерами, які можуть порадити, який тип професійних навичок відсутній у колективі. Крім кваліфікованого персоналу для роботи над проєктом, постачальник програмних послуг також повинен ретельно продумати технічні проблеми в розробці програмного забезпечення.

Найбільша проблема, як правило, виникає при адаптації останніх тенденцій ринку до старих застарілих систем. Для кваліфікованої команди розробників робота з інноваційними інструментами програмування – це не лише шанс практикувати щось нове і неапробоване, але й експериментувати та створювати унікальні продукти. Однак старіші застарілі системи та інтерфейси, які стають зайвими, можуть затримати весь проєкт, оскільки команда буде постійно вдосконалювати, підтримувати та виправляти їх, а не прогресувати. Поки мова йде про отримання максимально можливого доходу, здається дивним, що підприємства ще не позбулися старих спадкових систем.

Хоча існує чітка тенденція до вдосконалення та оновлення програмно-апаратних систем, багато зацікавлених сторін вважають за краще

дотримуватися старих застарілих систем та інтерфейсів тому що у них вкладаються кошти.

Визначальні характеристики портфелю проєктів об'єднуються, щоб побудувати робоче середовище, в якому фінансова та соціальні ставки настільки високі, що є зрозумілим бажання передбачити та проаналізувати майбутні наслідки, пов'язані з переслідуванням конкретної цілі.

Література, що звертається до цього питання, суттєво збігається з думкою про те, що воно стосується ризику та невизначеності. «Весь сенс існування управління проєктами є видалення, або істотне зменшення невизначеності щодо задоволення конкретних цілей» [30].

Таким чином, персоналу, відповідальному за прийняття рішень потрібно приділяти значну увагу методам і моделям, які призначені для визначення, оцінки та управління ризиками та невизначеностями.

Перша проблема цього питання полягає у визначенні математичного апарату необхідного для вибору моделей та методів, необхідних для передбачення та вирішення проблем управління портфелем проєктів. Друга проблема цього питання полягає у визначенні чи відрізняється ризик від невизначеності і, якщо так, то яким чином. Існує реальна небезпека, що у процесі прийняття рішень не розглядається ціла низка потенційно важливих питань, пов'язаних з ризиком або невизначеністю.

Аналізуючи літературні джерела [64] виявлено ключові відмінностей між ризиком та невизначеністю – табл. 1.3. Маючи різні види класифікацій проблем, виникаючих в процесі управління портфелем проєктів та класифікацію типів ймовірностей, застосованих для процесу управління, спробуємо розробити відповідність між ними – табл. 1.4. Найважливішим результатом цієї відповідності є формування классу проблем, які виникають у разі невизначенності та не можуть бути враховані стохастично. Також данна відповідність допоможе краще розуміти необхідність того чи іншого математичного апарату для вирішення поставлених проблем.

Треба мати на увазі, що рішення щодо структури портфеля спирається на попередню оцінку даних, процес здійснюється в умовах невизначеності, спричиненої невідомими майбутніми результатами.

Крім того, величезна складність і ненормальність на фінансових ринках робить стохастичний підхід менш придатним. Це є результатом того, що по перше вимірювана невизначеність – ризик. При роботі з класичними визначеннями ризику – визначається в детерміністичних або стохастичних умовах, і в результаті отримуємо чіткий (на відміну від нечіткого) набір чисел і чітких значень. По друге – нечітка невизначеність [2].

Таблиця 1.3 – Типи ймовірностей (Джерело – власна розробка)

Ризик / невизначеність	Типи ймовірностей	Характеристики
1	2	3
Ризик	Апріорна ймовірність	Є можливість назначити об'єктивні ймовірності для відомого діапазону ймовірних подій на основі імовірнісного розподілу.
	Статистична ймовірність	Є можливість назначити об'єктивні ймовірності для відомого діапазону ймовірних подій на основі емпіричних або статистичних даних стосовно аналогічних подій у минулому.
Невизначеність	Суб'єктивна ймовірність	Немає можливості назначити об'єктивні ймовірності для відомого діапазону ймовірних подій через нестачу необхідних даних. Рішення приймаються суб'єктивно з огляду на практику попередніх проєктів.
	Соціальна ймовірність	Немає можливості визначити ймовірності через відсутність чіткого діапазону ймовірних подій. Майбутнє непередбачуване через його соціальну складову.

Таблиця 1.4 – Класифікація проблем за типами ймовірностей (Джерело – власна розробка)

Типи ймовірностей	Проблеми
1	2
Апріорна ймовірність	1. Недієві методи оцінки; 2. Невизначено шляхи обміну інформацією між проєктами у портфелі та періодичність обміну.
Статистична ймовірність	1. Нечіткі ролі та обов'язки; 2. Відсутність контролю процесу управління; 3. Відсутність самонавчання.
Суб'єктивна ймовірність	1. Часті зміни у структурі управління проєктом (портфелем); 2. Підбір персоналу;
Соціальна ймовірність	1. Відсутність мотивації; 2. Нестача інформації по проєктам.

Використання апарату нечіткої логіки може виявитися цінним інструментом управління ризиками та невизначеністю в управлінні портфелем проєктів. Відображаючи властивості нечіткості та невизначеності, нечітка логіка дозволяє моделювати реальні умови, коли точна та однозначна інформація обмежена або відсутня. Застосування цього математичного підходу дозволяє враховувати різні рівні нечіткості, що присутні у прийнятті рішень, і використовувати нечіткі множини та правила для моделювання та аналізу ризику та невизначеності.

Використання апарату нечіткої логіки дозволяє управляти складністю та випадковістю, які часто супроводжують процес управління портфелем проєктів. За допомогою нечітких моделей можна врахувати суб'єктивність та неоднорідність інформації, а також узгоджувати різні точки зору та експертні оцінки. Враховуючи різні ступені приналежності до кожного ризику чи ситуації невизначеності, нечітка логіка дозволяє отримати більш гнучкі та реалістичні результати, які відображають реальний стан справ.

Крім того, апарат нечіткої логіки може використовуватися для моделювання та аналізу ситуацій, де існує неповна або неоднозначна інформація про взаємозв'язок між ризиками та їх наслідками. Використання нечітких правил та нечітких інтервалів дозволяє управляти ризиками та невизначеністю, враховуючи взаємозв'язки та взаємовплив різних факторів.

Таким чином, застосування апарату нечіткої логіки в управлінні портфелем проєктів дозволяє ефективно керувати ризиками та невизначеністю, враховуючи важливі характеристики цього середовища. Цей підхід дозволяє уникнути детерміністичного мислення та врахувати різноманітність факторів, що впливають на проектні рішення, сприяючи більш об'єктивному та збалансованому управлінню портфелем проєктів у умовах ризику та невизначеності.

1.3 Сучасні інноваційні технології управління IT-проєктами: зміст та етапи застосування

Скорочено оглянувши варіанти застосування різноманітних математичних моделей і методів, а також нечіткої логіки для вирішення задач або в рамках різноманітних підзадач, які постають безпосередньо перед менеджером проєктів, є сенс перейти до розгляду процесу управління проєктів у більш загальному сенсі, а саме перейти на рівень узагальнюючих структур управління процесом – фреймворками. Аналізуючи сучасні тенденції у використанні фреймворків управління проєктів, було виділено набір фреймворків для розгляду та аналізу стосовно використання в них нечіткої логіки.

Першим розглянемо фреймворк під назвою **SAFe** (Scaled Agile Framework). На його основі було реалізовано програмний продукт, задіяний на ринку з 2011 року [91]. Цей продукт охоплює досить велике коло задач у

рамках управління проєктами, але більше спирається на управління через досягнення стратегічних цілей та задоволення великої кількості ключових показників ефективності, найвагомішими з яких є показники пов'язані з задоволеністю клієнта.

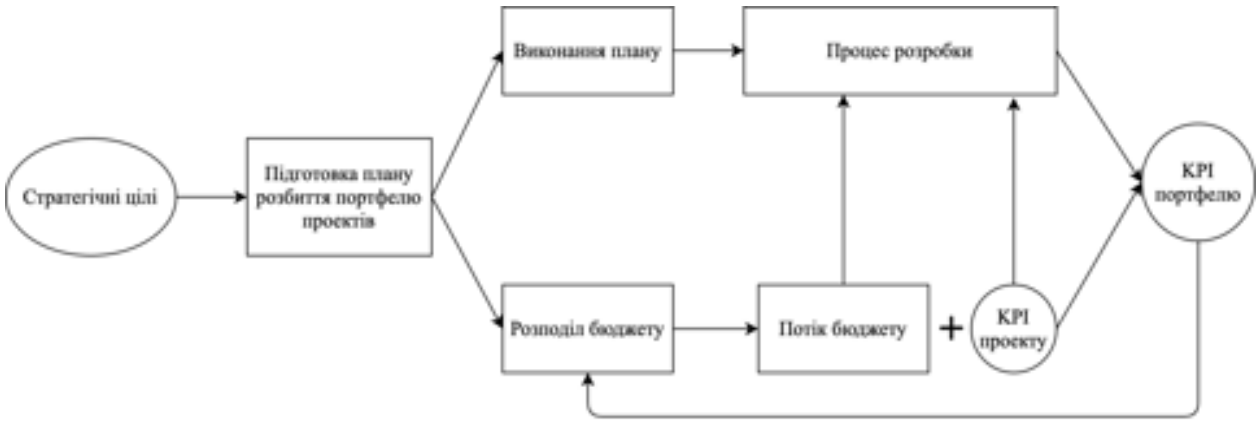


Рисунок 1.14 – Фреймворк SAFe у скороченій формі (Джерело – [91])

Не дивлячись на перспективність розвитку SAFe та на об’єм задач, в яких можна застосувати нечітку логіку в рамках цього фреймворку, застосування нечіткої логіки ускладняється наступним набором проблем – рис. 1.14:

- Фреймворк націлений на розгляд процесу управління проєктами на більш високому рівні, що тягне більшу доцільність застосування статичних методів, ніж методів нечіткої логіки;
- До уваги береться досить багато моментів, пов'язаних з задоволеністю клієнтів та стейкхолдерів, що наводить на думку про націленість фреймворку на задоволеність зацікавлених осіб більше, ніж підвищення ефективності управління;
- Не враховує особливостей управління та виконання IT-проєктів;
- Є закритим, деталі реалізації та конкретні деталі процесу не розголошуються, що робить неможливим його розширення ззовні.

Наступним розглянемо **DAD** (Disciplined agile delivery). Цей фреймворк надає гібридні та гнучкі підходи до керування та покращення процесу

постачання IT-рішень [26]. Розвивався на основі відомої технології SCRUM. Але є дві основні відмінності:

- В першу чергу ставить персонал, адже вважається, що якість постачаємого продукту на пряму залежить від якості комунікації та робочого середовища;
- Є масштабованим, що спрощує процес управління багатьма проєктами одночасно.

Відповідно до DAD життєвий цикл кожного проєкту розподіляється на наступні фази:

1. *Початок:* ціллю даної фази є ініціація та фіксація границь проєкту.
2. *Конструювання:* на цій фазі виконується реалізація раніше запланованого рішення інкрементним шляхом.
3. *Передача:* виконуються процеси постачання фінальної версії продукту зацікавленим особам та подальша підтримка у випадку необхідності.

Провівши аналіз поточної інформації можна зробити висновок, що застосування нечіткої логіки в даному фреймворку малодоцільне, адже в першу чергу ставляться проблеми соціальні, а найбільш вагомий етап конструювання проєкту не розподіляється на більш менші фази.

Останнім розглянемо Stage-Gate фреймворк [101]. Ефективно застосовуючи цей фреймворк, з'являється можливість керувати розробкою та запуском стійкого потоку успішних нових продуктів.

Особливість фреймворку є розподілення процесу протікання проєкту не тільки на окремі фази, а і на границі між цими фазами – рис. 1.15. Існування границь дає покращену можливість керувати зразу багатьма проєктами, адже є місця, спеціально виділені під прийняття рішень щодо процесу. Також ще однією перевагою є його статус "галузевого стандарту". Фреймворк є відкритим і не створює перешкод для створення його програмної реалізації. Також можна відмітити структуру фреймворку, яка робить досить простим процес його видозмінення, адаптації та модифікації.

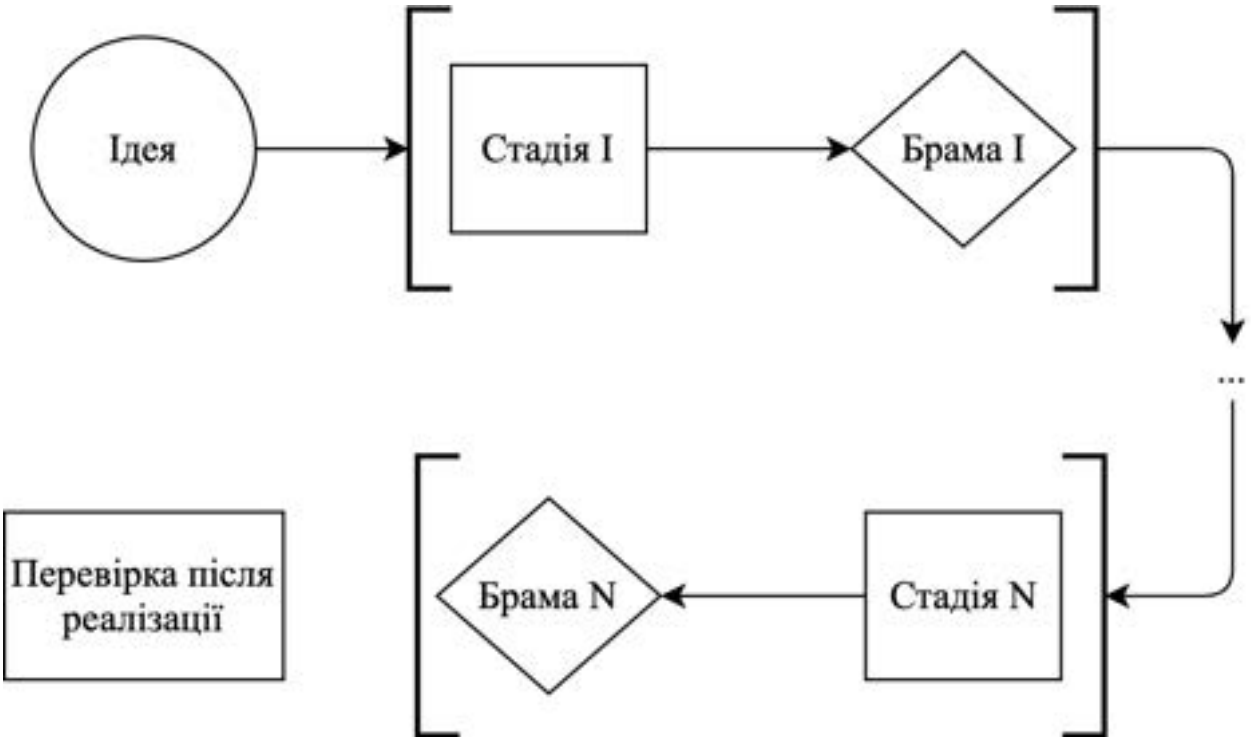


Рисунок 1.15 – Stage-Gate фреймворк у скороченій формі (Джерело – [101])

Stage-Gate фреймворк – це як концептуальна, так і операційна модель для переміщення нового продукту від ідеї до запуску. Це принцип управління процесом розробки нових продуктів, спрямованих на підвищення загальної ефективності та досягнення стратегічних цілей компанії. Хоча концептуально фреймворк досить простий, як побачимо пізніше, поглиблення в конкретні етапи проєкту ускладнює ситуацію. Stage-Gate фреймворк не є програмним продуктом, але його концептуальна простота та відкритість не ставить перешкоди при створенні конкретної програмної реалізації. Також, як буде обговорено далі, фреймворк дуже зручний для адаптації під конкретні потреби та особливості предметної області.

Зі сторони виглядає, що ідея розподілу процесу загалом на менші підпроцеси не нова. Але ідея фреймворку не тільки в розподілі. Між кожним етапом додатково вводиться границя для прийняття рішень при переході між етапами, що якісно відрізняє процес формальною наявністю точки прийняття рішень. Також додатковим позитивним ефектом у виділенні границі є

можливість винести обчислення безпосередньо з етапу на границю. Більш чіткий розподіл дозволяє кожному окремому етапу сфокусуватися на зборі інформації та делегувати її обробку границі.

Досить багато компаній робили спроби формалізувати процеси розробки нових продуктів у вигляді документованого процесу. Індивідуальні компанії можуть посилатися на свої системи різними іменами, а на папері вони здаються унікальними для процесів цієї компанії. На практиці, однак, спостерігається дуже сильний паралелізм між різними стадіями та підходами до їх розподілення.

Хороша аналогія для описання процесу це виробничий процес на заводі по виробництву деякої фізичної продукції. Безумовно, спосіб покращити продукцію – зосередити увагу на самому процесі – піддати контролю варіативність процесу. Процес поділяється на декілька етапів або робочих станцій. Між кожною робочою станцією є пункт контролю якості або границя. Набір результатів визначається для кожної границі, а також набір критеріїв якості, які повинен пройти продукт перед переходом на наступну робочу станцію. Етапи – це виконана робота; границі забезпечують перевірку якості.

Stage-Gate фреймворк уособлює собою подібну модель для управління процесом розробки. Фреймворк поділяє процес на заздалегідь визначений набір етапів, самі по собі складаються з групи прописаних, пов'язаних і часто паралельних заходів. Наприклад, етап "Перевірка" може містити перелік обов'язкових або необов'язкових заходів, таких як тестування прототипу або тестування з клієнтами. Типічний приклад фреймворку представлено на – рис. 1.16.

Границя є точкою переходу з етапу на етап. Можна сказати, що границі у деякій мірі керують процесом. Кожна границя характеризується набором результатів попереднього етапу та набором критеріїв переходу на наступний етап. Вхідні дані є результатами, які лідер проєкту доставляє на границю. Критерії переходу – пункти, на яких буде оцінюватися успішність попереднього етапу проєкту. Базуючись на цих критеріях приймається рішення

щодо переходу на наступний проєкт, завершення проєкту, відстрочці проєкту або його переробці на користь інших проєктів.

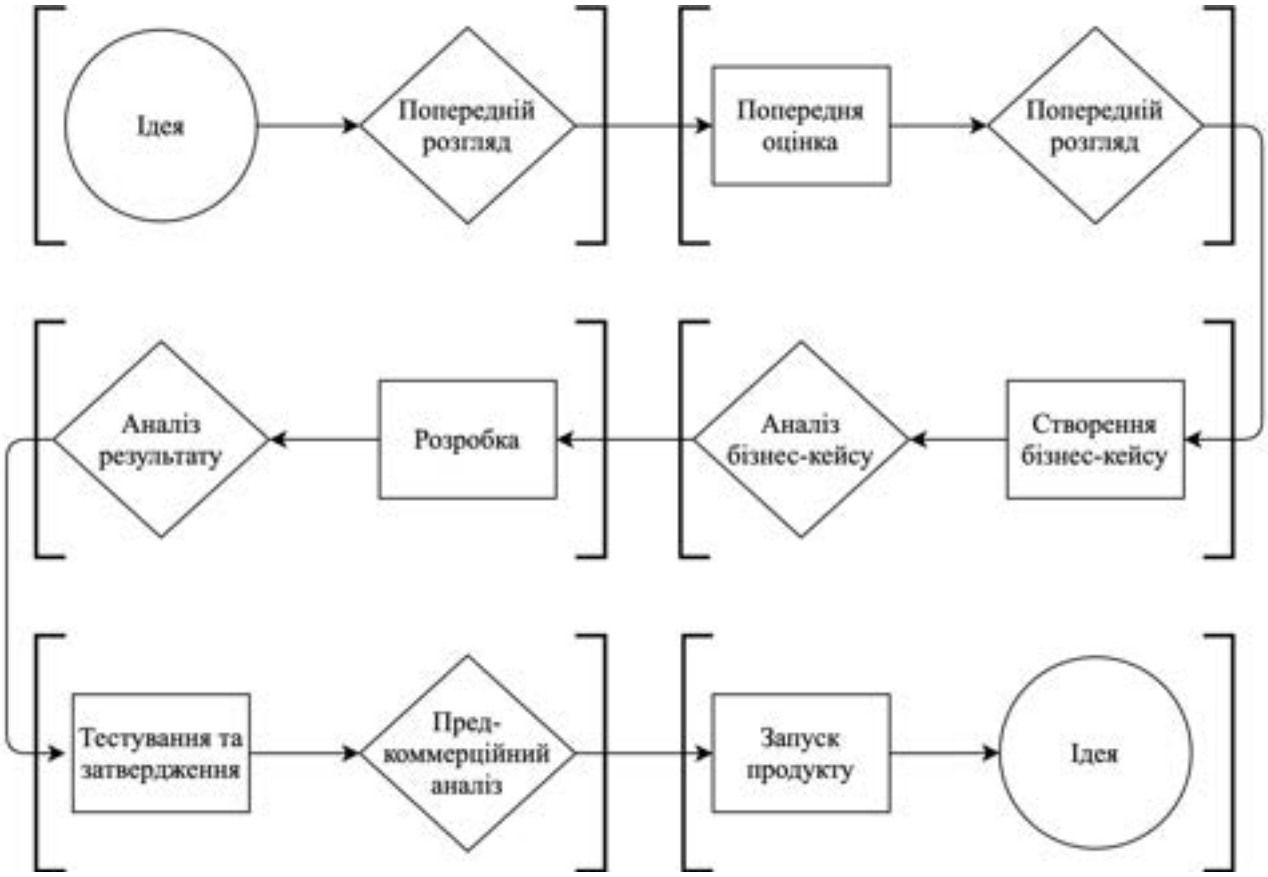


Рисунок 1.16 – Приклад Stage-Gate фреймворку (Джерело – [101])

Базова структура фреймворку сприяє усуненню частини обмежень, які накладаються на процес управління. А ті обмеження, які усунути не вдається одразу, ліквідуються за рахунок можливості адаптувати фреймворк під власні потреби. Більш детальніше розглянемо можливості усунення обмежень управління ІТ-проєктів.

Проблема візуалізації проєкту, або, якщо сказати точніше, візуалізації його прогресу залежить від ступеню інтеграції у процес управління. Але загалом, проблема вирішується за рахунок розширення етапів розробки та тестування в рамках фреймворку таким чином, щоб було простіше слідкувати за ходом процесу розробки. Доречі, такого роду удосконалення сприяє і вирішенню проблеми з гнучкістю та складністю процесу. Адже розширення

етапів розробки та тестування сприяє не тільки кращій візуалізації поточного стану проєкту, а і надає спроможність динамічно вводити нові або змінювати вже реалізовані функції, тримаючи процес видозмінення під контролем. Позбавлення проблеми невизначеності, як і проблеми візуалізації, залежить від ступеню інтеграції в процес. Базова структура фреймворку досить ефективно дозволяє зменшити ступінь невизначеності на ранніх етапах проєкту. Таким чином, Stage-Gate фреймворк є досить ефективним для описання процесу управління IT-проєктами.

Далі детальніше розглянемо використання Stage-Gate фрейморку в процесі управління проєктами. Також розглянемо недоліки та труднощі, виникаючі в процесі управління IT-проєктами при використанні Stage-Gate фреймворку.

Stage-Gate – це фреймворк, який дозволяє в рамках розглядаємої задачі створити і конвертувати модель ризиків у процес прийняття рішень стосовно цих ризиків [102]. З іншої сторони цей фреймворк призначений для швидкого та вигідного перетворення найкращих нових ідей організації в успішні прибуткові нові продукти. Його використання, як правило, створює культуру удосконалення процесу управління проєктами, націлених на створення продуктів, а саме лідерства, продуктивності команди, орієнтації на споживача та ринку, надійності рішення, швидкості та якості. Розглянемо детальніше завдяки чому Stage-Gate фреймворк дозволяє досягти підвищення ефективності управління проєктами. А саме розглянемо за рахунок чого створюється культура удосконалення процесу управління проєктами.

Орієнтація на клієнта. Це один за найважливіших ключів до успіху та рентабельності проєкту, що розробляється. Його мета – зосередити увагу на розробці та постачанні нових продуктів, які вирішують основні проблеми замовників та пропонують клієнту вагомі пропозиції щодо цінності проєкту, що розробляється. Досягається це за рахунок того, що концепт орієнтації на клієнтів вбудований у весь процес Stage-Gate фреймворку. Починаючи прямо на початку проєкту, ідеї, створені як внутрішньо, так і ззовні, перевіряються за

замовником. На кожному етапі триває постійна взаємодія із замовником, наприклад: тестування концепцій, раннє та швидке тестування прототипу для забезпечення того, щоб дизайн залишався вірним тому, що хоче і за що заплатить замовник. Це гарантує, що новий продукт/послуга пропонує замовнику нові та унікальні переваги, які забезпечують кращу цінність проєкту клієнту, ніж пропонує конкуренція.

Попередні заходи. Успіх чи невдача нового продукту часто визначається ще до того, як він вступає у стадію розробки. Самі найважливіші заходи, що визначають особливості та переваги нового продукту, дадуть клієнту зрозуміти цінність цього продукту. Компанії, які практикують використання попередніх заходів, досить часто скорочують час виходу на ринок. Це робиться шляхом покращеного процесу визначення продукту, забезпечення правильності специфікацій продукту, підтвердження дизайну продукту на початку процесу його розробки та мінімізації ціни змін в останню хвилину. Такі компанії також уникають поспішати в розробці продуктів, думаючи, що вони найкраще знають, що потребує ринок. Натомість вони вживають критичних кроків для того, щоб забезпечити чітко визначену ціль та направленість продукту, які будуть затверджені з ґрунтовним внеском у процес маркетингу цього продукту. Усі попередньо розглянуті переваги концептуально вбудовано в Stage-Gate фреймворк, що дозволяє значно зменшити впливи пізнішого прийняття рішень.

Фіксовані точки прийому рішень. Загальні симптоми того, що система управління проєктами не працює належним чином: занадто багато проєктів, для яких не вистачає ресурсів; процес розробки рухається занадто повільно; проєкти ніколи не згортаються. У свою чергу, компанії, які цих симптомів позбавлені, гарантують, що вони мають чіткі точки (або ворота) прийняття рішень та чітко визначені особи, які приймають ці рішення. Основна концепція Stage-Gate фреймворку полягає у існуванні точок прийняття рішення протягом усього процесу, коли керівництво та проєктні команди можуть зібратися, щоб переглянути проєкт та його переваги для майбутнього фінансування. Кожен проєкт переглядається на відповідність чітким критеріям, а потреби в ресурсах

проєкту порівнюються з іншими конкуруючими проєктами, що знаходяться у процесі розробки. Рішення, засновані на фактах, приймаються на цих засіданнях. Ті, хто приймає рішення – це люди, які розпоряджаються ресурсами і можуть приймати рішення щодо розподілу цих ресурсів, необхідних командам проєкту для просування проєкту вперед. Це веде до вчасного прийняття рішень щодо затвердження або згортання проєктів.

Процес Stage-Gate заснований на переконанні в тому, що будь-яка продукція починається з ідеї і закінчується успішним запуском продукту на ринок. Stage-Gate фреймворк розбиває часто складний і хаотичний процес розробки продукту від початку до запуску на менші етапи (де проводяться проєктні заходи) та перехідні брами (де проводяться оцінки й приймаються рішення Go/Kill). У повному обсязі Stage-Gate містить у собі реалізацію різних видів заходів − попередніх, розробки (технічна, маркетингова та операційна) та комерційної діяльності (вихід на ринок та збір інформації після запуску) – об’єднаних у єдиному бізнес-процесі [102].

Керівник проводить проєкт через кожен етап фреймворку. Кожен етап призначений для збору конкретної інформації, яка допоможе перевести проєкт на наступний етап або перехідні брами. Конкретний етап визначається типом діяльності всередині нього. Ці заходи проводяться паралельно, що дозволяє проєктам швидко рухатися до завершення, і пов'язані між собою. Основна направленість заходів йде на збір інформації та поступове зменшення невизначеності та ризику. Кожен наступний етап стає дорожчим від попереднього і підкреслює важливість збору додаткової інформації для зменшення невизначеності. Результати цього інтегрованого аналізу стають набором результатів, які використовуються у перехідних брамах для прийняття рішень.

На цих перехідних брамах приймається рішення про те, чи продовжувати інвестування проєкту. Вони виконують роль пунктів контролю якості за трьома цілями: забезпечити якість виконання, оцінити бізнес-обґрунтування та

затвердити план та ресурси проєкту. Кожна брама структурована за наступним способом:

1. Результати етапу;
2. Критерії прийняття рішень;
3. Рішення.

Результати. Керівник проєкту та група розробників надають відповідальним особам результати діяльності, завершеної на попередньому етапі.

Критерії прийняття рішення. Кожен проєкт оцінюється відповідно до чітко визначеного набору критеріїв успіху. Критерії повинні бути надійними, щоб швидше визначити виграшні проєкти. Автентичний Stage-Gate фреймворк, як правило, включає шість перевірених критеріїв:

- Стратегічна відповідність;
- Продуктова та конкурентна перевага;
- Привабливість ринку;
- Технічна можливість;
- Можливості об'єднання / синергії з іншими проєктами;
- Фінансова винагорода / ризик.

Рішення. Приймається рішення (Go/Kill/Hold/Recycle). Якщо було прийнято рішення щодо продовження розробки проєкту, то нові ресурси на розробку зобов’язуються бути виділеними. План дій на наступний етап затверджено. Визначено перелік результатів та дату наступної зустрічі для прийняття рішення.

Далі розглянемо класичні етапи та брами Stage-Gate фреймворку.

Брама 1 – Оцінка ідеї (Попередній розгляд):

Це перше рішення щодо залучення ресурсів до проєкту: проєкт народжується в цей момент. Оцінка ідеї є "легкою брамою" і становить собою оцінку проєкту за кількома якісними критеріями такими як стратегічна відповідність цілям компанії, необхідність проєкту в рамках поточної або майбутньої фінансової ситуацій компанії, наявність технічних можливостей для

його виконання та привабливість ринку до цього проєкту, перевага проєкту перед конкурентами, можливість використання ресурсів компанії та відповідність політиці компанії [38]. Фінансові критерії, як правило, не є частиною цієї брами.

Етап 1 – Оцінка масштабу (Попередня оцінка):

Цей перший та недорогий етап первинної оцінки має на меті визначити технічні та ринкові переваги проєкту. Оцінка масштабу – це швидка оцінка обсягу проєкту – тут мало або взагалі не робиться первинних досліджень. Оцінка масштабу часто проводиться менше ніж один календарний місяць, і на його здійснення потрібно, як правило, не більше 10-20 робочих людино-годин. Він включає такі заходи, як:

- попередня оцінка ринку,
- попередня технічна оцінка,
- попередня бізнес оцінка.

Брама 2 – Підтвердження оцінки (Повторний розгляд):

Проєкт піддається другій і дещо жорсткішій оцінці на брамі підтвердження оцінки. Ця брама, по суті, є повторенням першої брами: проєкт переоцінюється з урахуванням нової інформації, отриманої з етапу оцінки масштабу. Якщо рішення буде ухвалено під час процесу підтвердження оцінки, проєкт переходить у більш важку стадію оцінки витрат. Окрім якісних критеріїв, використовуваних у брамі оцінки ідеї, на брамі підтвердження оцінки оцінюється фінансова віддача, але лише за допомогою швидкого та простого фінансового розрахунку (наприклад, за періодом окупності).

Етап 2 – Побудова бізнес-кейсу:

Бізнес-кейс відкриває дорогу до подальшого розвитку продукту. Цей етап є етапом детального дослідження, який чітко визначає місце продукту в межах компанії та підтверджує привабливість проєкту до великих витрат. Це також важливий етап для виконання багатьох попередніх заходів. Основні заходи включають:

- Дослідження ринку, включаючи дослідження думки замовника, щоб визначити потреби, бажання та потенційні переваги замовника, які допоможуть визначити продукт;
- Конкурентний аналіз;
- Тестування концепції;
- Детальна технічна оцінка зосереджена на технічній доцільності проєкту;
- Оцінка виробничих можливостей компаній для виконання проєкту;
- Детальний бізнес та фінансовий аналіз разом з аналізом чутливості, щоб переглянути можливі ризики виконання проєкту.

Результат цього етапу – бізнес-кейс для проєкту: погоджене визначення продукту, розроблено обґрунтування та детальний план проєкту. Етап побудови бізнес-кейсу передбачає значно більше зусиль, ніж етап оцінки масштабу, і найкращим чином ним займається команда, що складається з основної групи майбутньої команди проєкту.

Брама 3 – Перехід до розробки (Аналіз бізнес-кейсу):

Це остання брама перед етапом розробки, останній момент, коли проєкт може бути ліквідовано перед тим, як на нього буде витрачено велику кількість ресурсів. Після того, як мине брама переходу до розробки, фінансові зобов'язання стають значними. Іншими словами, можна сказати що, брама переходу до розробки означає "піти на великі витрати". Ця брама також підтверджує визначення проєкту. Якісна сторона цієї оцінки включає перегляд кожного з заходів на етапі побудови бізнес-кейсу та перевірку того, чи була їх якість виконання надійною, а результати позитивними. Далі, проєкт перевіряється ще раз через набір критеріїв «Маємо реалізувати/Варто реалізувати», які були сформовані на етапі побудови бізнес-кейсу, але цього разу із значно більшою суворістю та з використанням більш надійних даних. Нарешті, оскільки великі зобов'язання щодо витрат є результатом рішення цієї брами, результати фінансового аналізу також є важливою частиною цього екрану. Якщо рішення ухвалено, то підписується визначення продукту та

узгоджується план проєкту, який відображає шлях його подальшої розробки. Узгоджується повна команда проєкту, яку очолює лідер з повноваженнями.

Етап 3 – Розробка:

Перехід до етапу розробки свідчить про початок виконання плану розвитку продукту, підписаного на брамі переходу до розробки. Лабораторні випробування, внутрішні тести або альфа-тести забезпечують, щоб проєкт відповідав вимогам. Для більш тривалих проєктів численні проміжні етапи та періодичні огляди проєкту вбудовуються в план розвитку. Ці періодичні огляди не є брамами: тут не приймаються рішення щодо плану розвитку проєкту; скоріше ці контрольні пункти забезпечують підтримання безперебійного контролю та управління проєктами. На цих пунктах, зазвичай, проводяться широкі внутрішні тестування, альфа-тести або лабораторні тести. Наприкінці етапу розробки постачається внутрішній протестований прототип продукту. Акцент на етапі розробки робиться на технічну складову роботи. Але маркетингова та операційна діяльність також протікають паралельно. Наприклад, робота з аналізу ринку та зворотного зв’язку з клієнтами триває одночасно з технічною розробкою. Також готується оновлений фінансовий аналіз, а також вирішуються питання регуляторної та інтелектуальної власності.

Брама 4 – Перехід до тестування (Аналіз результату):

Ця брама націлена на перевірку прогресу розробки та відповідності потребам та вимогам замовника щодо продукту та проєкту. Роботи стосовно розробки переглядаються та перевіряються, гарантуючи, що робота була завершена якісно, і що розроблений продукт дійсно відповідає оригінальному визначенню, на брамі переходу до розробки. Також переглядаються економічні показники проєкту шляхом перегляду фінансового аналізу на основі нових і більш точних даних. Проходить процес затвердження планів тестування та валідації для наступного етапу. Детальні маркетингові плани та операції переглядаються на предмет найскорішого майбутнього виконання.

Етап 4 – Тестування:

Цей етап перевіряє та підтверджує життєздатність проєкту: сам продукт, виробничий процес, ступінь прийняття замовником та економічні показники проєкту. Також розпочинається досить велика зовнішня перевірка товару замовником. На етапі тестування проводиться ряд заходів:

- Внутрішні тести продукту: розширені лабораторні тести або альфа-тести, щоб перевірити якість продукту та продуктивність в контрольованих умовах або в лабораторних умовах.
- Випробування користувачем: перевірити, чи продукт функціонує в реальних умовах використання, а також оцінити реакцію потенційних клієнтів на продукт - розрахувати вірогідність використання та покупки.
- Пробне, обмежене або пілотне виробництво продукту: для випробування, налагодження та підтвердження виробничого чи операційного процесу, а також для визначення більш точних витрат на виробництво/експлуатацію та пропускну здатність.
- Попереднє тестування перед виходом на ринок або пробний продаж: для оцінки реакції клієнтів, розрахування ефективності плану запуску та визначення очікуваної частки ринку та доходів.
- Переглядається бізнес-фінансовий аналіз: перевірити тривалу ділову та економічну життєздатність проєкту на основі нових і більш точних даних про доходи та витрати.

Іноді етап тестування дає негативні результати. У такому разі виникає необхідність повертатися до етапу розробки.

Брама 5 – Перехід до запуску (Предкомерційний аналіз):

Ця заключна брама відкриває двері до повної комерціалізації – запуск на ринок та запуск виробничих операцій. Це заключний момент, коли проєкт все ще може бути ліквідовано. Брама переходу до запуску зосереджена на перевірці якості діяльності, проведеної на етапі тестування. Критерії проходження цієї брами в основному зосереджені на очікуваній фінансовій віддачі та доцільності планів запуску. Операційні та маркетингові плани переглядаються та затверджуються для впровадження на наступному етапі.

Етап 5 – Запуск продукту:

Цей останній етап включає в себе реалізацію як плану запуску маркетингу, так і плану виробничих операцій. Враховуючи продуманий план дій та факт підкріплення його відповідними ресурсами, і враховуючи будь-які непередбачені події, процес запуску продукту буде чітким, послідовним та передбачуваним.

Пост-етап – Огляд після запуску (Аналіз проєкту):

У якийсь момент після комерціалізації (часто після 6-18 місяців) проєкт повинен бути припинений. Команда розпущена, і продукт стає добре протестованим та перевіреним замовником. Після цього продукт стає ще одним надійним продуктом у товарній лінійці компанії. На цьому етапі переглядається ефективність проєкту та продукту. Останні дані про доходи, витрати, видатки, прибутки та терміни порівнюють з прогнозами етапів розробки та запуску продукту на оцінку ефективності. Нарешті, проводиться аудит – критична оцінка сильних та слабких сторін проєкту, чого можна дізнатися з цього проєкту та як краще зробити наступні, враховуючи цю інформацію. Цей огляд сигналізує про закінчення проєкту. Важливо мати на увазі, що проєктна команда та керівник залишаються відповідальними за успіх проєкту протягом цього етапу.

Stage-Gate фреймворк також можна використовувати для детального визначення операцій за розробкою програмного забезпечення, які зазвичай виконуються на кожному етапі. Крім того, він визначає критерії виходу, на підставі яких проєкту дозволяється завершити один етап і перейти до наступного.

Спеціалізація Stage-Gate фреймворку значно залежить від організації, методики розробки та типу проєкту, що розробляється. Продемонструємо, як Stage-Gate фреймворк координується з типовим процесом поетапного контролю, який орієнтується на програмний продукт.

1. Виявлення.

a. *Основні операції:* виявлення ринкової можливості, оцінка технічної прийнятності, розробка попередньої економічної моделі.
b. *Основні критерії виходу:* узгодження попередньої економічної моделі.

2. Аналіз вимог.
    a. *Основні операції:* формування концепції продукту, розробка ринкових вимог.
    b. *Основні критерії виходу:* узгодження концепції продукту, узгодження маркетингових вимог.
3. Планування.
    a. *Основні операції:* розробка планів та вимог, розробка оцінки бюджету, розробка остаточної економічної моделі.
    b. *Основні критерії виходу:* узгодження планів розробки, узгодження бюджету, узгодження остаточної економічної моделі.
4. Розробка.
    a. *Основні операції:* основний життєвий цикл розробки продукту, розробка маркетингового і оперативного планів, розробка остаточного плану тестування.
    b. *Основні критерії виходу:* узгодження плану випуску програмного продукту, узгодження маркетингового і оперативного планів, узгодження плану тестування.
5. Тестування.
    a. *Основні операції:* виконання остаточного плану тестування, виконання критеріїв випуску, ухвалення рішення про випуск.
    b. *Основні критерії виходу:* виконання плану впровадження.
6. Запуск.
    a. *Основні операції:* виконання плану впровадження, збір думки замовника і повідомлень про дефекти, відстеження економічних результатів.
    b. *Основні критерії виходу:* остаточний вихід з фреймворку.

Необхідно зауважити, що з точки зору управління IT-проєктами, Stage-Gate фреймворк створює деякі проблеми. Одна з проблем полягає в тому що спочатку цей фреймворк розроблявся для процесів виготовлення обладнання та споживчих продуктів, не маючих відношень до процесу розробки IT-проєктів. Хоча основна структура таких процесів залишається корисною, її необхідно адаптувати до специфіки процесів розробки IT-проєктів, щоб вона працювала з такою же ефективністю, як і при розробці інших видів продуктів.

Наступна проблема, одна з найбільших, полягає в тому, що процес програмної розробки найчастіше представляється у вигляді одного етапу. Звичайно в більш нових версіях фреймворку робляться обмовки щодо можливості повертатися до деяких попередніх етапів у фреймворці, але це не враховує циклічності цього процесу в повній мірі. Операції, які виконуються в процесі програмної розробки, займають від 75% до 90% часу від загального об'єму робочих процесів. Було би доцільніше враховувати циклічність цього процесу або мати декілька проміжних точок на цьому етапі, що дозволить приймати рішення та проводити переоцінку частіше.

Інша стандартна проблема: етапи аналізу вимог і планування часто об'єднуються в один етап. Фактично це означає, що розподілення календарного плану стосовно ранніх етапів проєкту лежить у проміжку від 15% до 35% календарного часу проєкту.

Учасникам проєкту, не пов'язаним з технічною стороною, часто доводиться пояснювати, які операції по розробці програмного забезпечення повинні бути завершені для отримання осмисленого уявлення про проєкт перед переходом на етап програмної розробки. І з досить широким діапазоном розподілу календарного часу перед цим, є досить велика вірогідність перейти до етапу програмної розробки без всієї необхідної інформації, або раніше необхідного часу [71].

Як бачимо, Stage-Gate фреймворк є досить ефективним інструментом для розробки фізичної продукції, адже саме для цього цей фреймворк спочатку і розроблявся. Але коли фреймворк адаптується до процесу розробки

програмного забезпечення, виникає низка проблем, пов'язаних з разючою відмінністю підходів до процесу.

Ці відмінності були менш разючими за часи використання послідовних етапів розробки програмного забезпечення, що приводило до мінімальних змін у структурі Stage-Gate фреймворка для адаптації до процесу. Але прихід набору Agile технологій розробки програмного забезпечення звів до мінімуму практики використання послідовних етапів розробки.

Це, у свою чергу, призвело до ускладнення процесу адаптації Stage-Gate фреймворку до сучасних процесів розробки програмного забезпечення.

У життєвому циклі розробки програмного забезпечення є декілька моментів, заслуговуючих найбільшої уваги.

По-перше, необхідно акцентувати увагу на самому процесі.

По-друге – на якість програмного забезпечення та самого процесу. Agile процеси розробки програмного забезпечення – це ітеративна розробка, де вимоги змінюються відповідно до потреб замовника. Це допомагає в адаптивному плануванні та ітеративному розвитку. Вибір конкретної Agile моделі залежить від специфіки проєкту та процесів роботи з замовником [39].

Agile процес відповідає життєвому циклу розробки програмного забезпечення, який включає збір вимог, аналіз, проєктування, розробку, тестування, впровадження частково реалізованого програмного забезпечення з очікуванням зворотного зв’язку з замовником [94].

Рис. 1.17 зображає найбільш загальний життєвий цикл розробки програмного забезпечення в Agile процесі.

Agile процес в цілому схожий з Stage-Gate фреймворком, за виключенням чітко виражених брам, на яких приймаються рішення щодо продовження проєкту. Agile процес вимагає менше планування і розділяє задачі на невеликі підзадачі.

Цей процес від початку було призначено для короткотермінових проєктів з врахуванням специфіки командної роботи, яка відповідає життєвому циклу розробки програмного забезпечення.

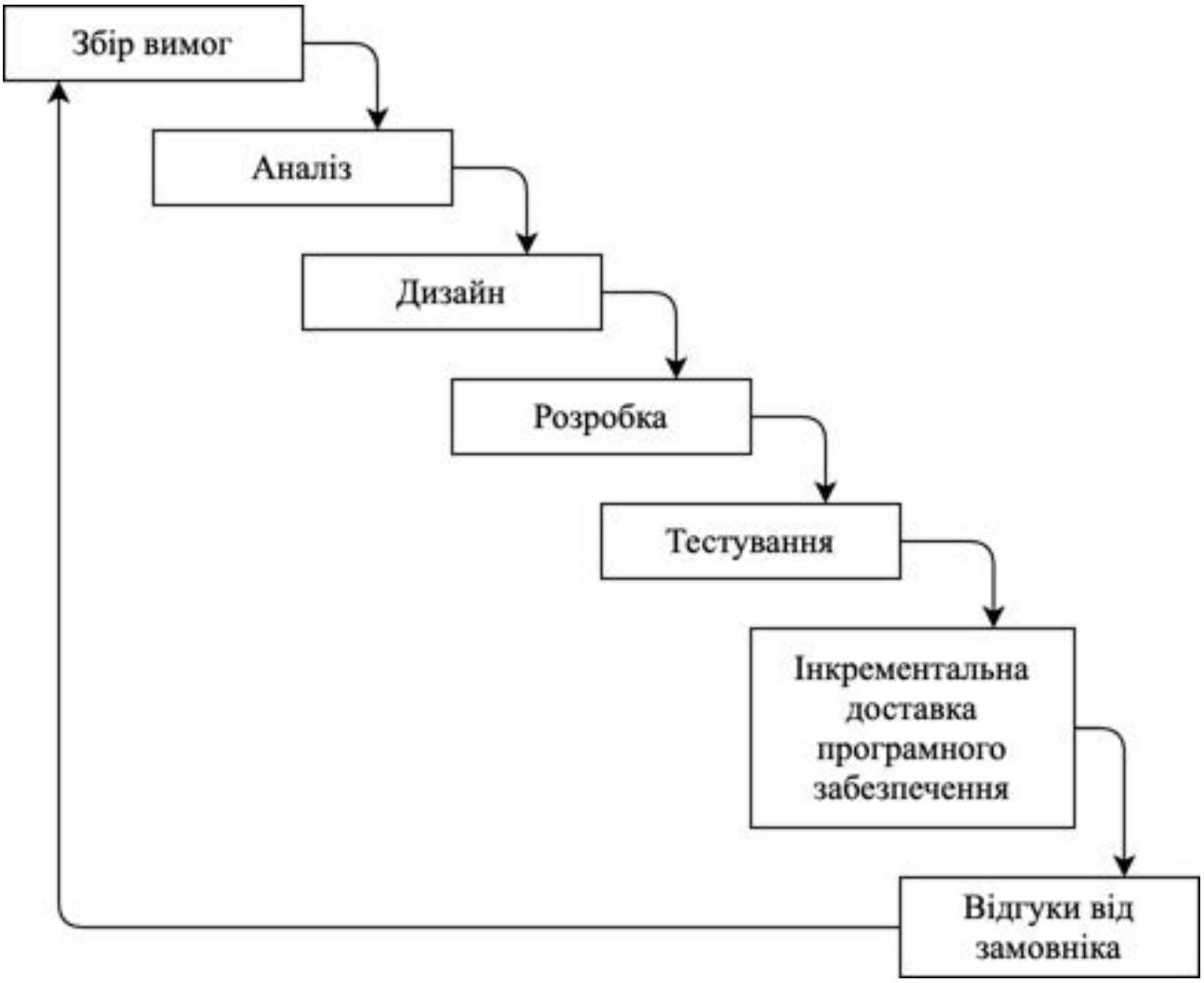


Рисунок 1.17 – Життєвий цикл Agile (Джерело – [94])

Але з часом почали розвиватися наслідні технології, які розширюють процес на проєкти та команди більших розмірів. Залучення замовника до процесу управління програмного забезпечення зменшує ризики, пов'язані з пізнішими етапами розробки програмного забезпечення, такими як впровадження на ринок та тестування на стороні замовника. Досягається це за рахунок того, що це процес, в якому можна вносити зміни динамічно відповідно до задоволеності та потреб замовника. Розглянемо основні принципи Agile процесу:

- *Ітеративність:* основна мета Agile програмних процесів − задоволення клієнтів, тому ці процеси зосереджені на виконанні вимог за одну або кілька ітерацій;

- *Модульність:* Agile процес розбиває всю систему розробки програмного забезпечення на керовані частини, які називаються модулями;
- *Тайм-боксинг:* оскільки Agile процес носить ітеративний характер, він вимагає часових обмежень для кожного модуля;
- *Приріст:* оскільки Agile процес носить ітеративний характер, він вимагає, щоб система розвивалася покроково, таким чином, щоб кожен приріст не залежав від інших, і так щоби всі прирости інтегрувалися в повну систему;
- *Адаптивність:* у зв'язку з ітераційним характером Agile процесу можуть виникати нові ризики, пов'язані з роботою з замовником, досить часто з'являється необхідність адаптувати процеси до нових ризиків та змінити потреби в реальному часі;
- *Спільність роботи:* оскільки Agile процес є модульним за своєю суттю, він потребує хорошої комунікації між командою з розробки програмного забезпечення, так як в кінці процесу розробки програмного забезпечення потрібно інтегрувати різні модулі.
- *Орієнтованість на людей:* по-перше, задоволення клієнтів є першочерговим завданням, по-друге, хороша команда з розробки програмного забезпечення також грає досить високу роль, адже підвищує продуктивність та якість процесу розробки програмного забезпечення.

Існує кілька технологій, за допомогою яких можливо реалізовувати Agile процеси. Кожна з цих технологій орієнтована на різні аспекти життєвого циклу розробки програмного забезпечення [69]. Деякі зосереджуються на практиці (екстремальне програмування, парне програмування), а інші зосереджуються на управлінні програмними проєктами (Scrum). Розглянемо технології, які найбільш широко використовуються в промисловості.

Екстремальне програмування (Extreme Programming, XP) – це метод спритної розробки програмного забезпечення, який найбільшою мірою базується на орієнтації на задоволення замовників. Для розробки програмного забезпечення в цій технології потрібна максимальна взаємодія з клієнтами. Він

розділяє весь життєвий цикл розробки програмного забезпечення на кілька коротких циклів розробки. Він володіє динамічною можливістю, включає в себе зміни або вимоги від замовників на будь-якій фазі життєвого циклу розвитку проєкту.

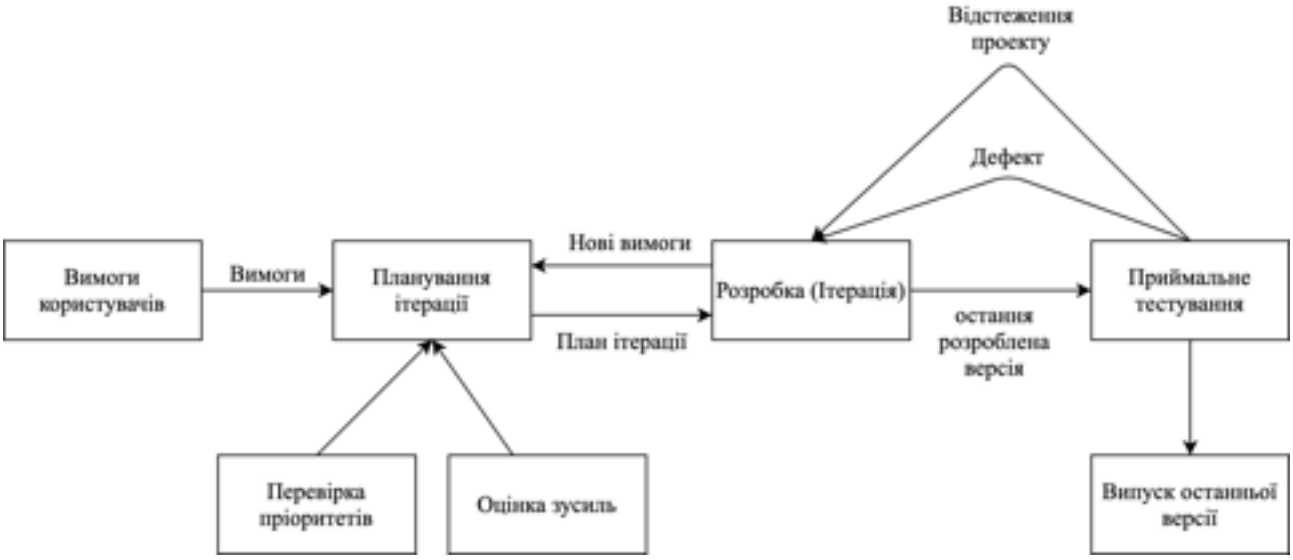


Рисунок 1.18 – Життєвий цикл XP (Джерело – власна розробка)

Наведена вище схема (рис. 1.18) показує повний метод розробки Agile процесу за допомогою методу XP. Екстремальне програмування починається зі збору вимог користувачів. Залежно від цих вимог весь процес розбиття поділяється на кілька невеликих циклів. Отже, наступним етапом є планування ітерацій, тобто визначення кількості циклів, визначення пріоритетності вимог та оцінка кількості зусиль, необхідних для виконання кожного циклу. Під час фази розробки можуть з’являтися нові вимоги користувачів і план ітерацій слід коригувати відповідно до цього. Наступним кроком є тестування останньої розробленої версії. Після кожної фази тестування слід проводити загальний огляд виконаних робіт.

Хоча інші Agile технології набули більшої популярності, досі можна бачити результати впливу впровадження XP технології на IT-галузь. А саме XP технологія запропонувала багато нових речей для розробників, таких як парне

програмування, розширений перегляд коду більш досвідченими спеціалістами, рефакторинг коду та відкриті робочі області.

Scrum – технологія agile процесу, яка базується на поступовому процесі розробки програмного забезпечення. У цій технології весь цикл розробки поділяється на серію ітерацій, де кожна ітерація називається спринтом – рис. 1.19. Максимальна тривалість спринту – 30 днів.



Рисунок 1.19 – Життєвий цикл Scrum (Джерело – власна розробка)

Метод починається зі збору вимог замовником, але цікаво, що не очікується, що всі вимоги повинні бути надані замовником на початку процесу розробки. Замовник може змінити свою думку в будь-який час під час розробки – можуть додавати нові функції, видаляти або оновлювати деякі існуючі функції. Наступним етапом є визначення пріоритетності вимог, і списку відомий як товарний запас продукту. Проводиться належне планування спринтів, тобто, скільки спринтів потрібно для розробки програмного забезпечення, тривалість спринту та які вимоги щодо товарного запасу повинні бути реалізовані у кожному спринті. Цей конкретний список відомий як товарний запас спринту. Під час кожного спринту щодня проводиться одна зустріч, на якій можна отримати зворотній зв'язок, скільки роботи було виконано та скільки роботи очікується виконати. Після кожного спринту, проводиться перегляд та аналіз цього спринту, щоб визначити, чи всі вимоги цього конкретного спринту вже виконані чи ні, і щоб сформувати список вимог,

які слід реалізувати в наступному спринті. Таким чином після кожного спринту отримуємо приріст функціональності програмного забезпечення.

Функціональна розробка (Feature Driven Development, FDD) – ключова перевага цього методу полягає в розробці домену програмного забезпечення до початку процесу програмної розробки. Метод починається зі збору потреб та вимог замовника та створення загальної моделі проєкту, адже модель дає чітке уявлення про сферу застосування програмного забезпечення. Наступним кроком є складання списку функцій для оцінки замовником. Потім декілька груп функцій складаються на основі їх доменів, тобто пов'язані функції об'єднуються в одну групу. Кожна група функцій призначається команді розробників, яку очолює один головний програміст. Останнім кроком є моделювання ітерацій, в яких розробляються функції з груп. Цей крок продовжується, поки всі функції не будуть успішно реалізовані – рис. 1.20.



Рисунок 1.20 – Життєвий цикл FDD (Джерело – власна розробка)

Канбан – цю технологію можна розглядати як систему візуалізації потоку роботи. Вона призначена для виявлення чітких обмежень на скільки елементів може бути виконано у кожному сегменті робочого процесу та часу виконання для завершення одного елемента. За допомогою вимірювання цих показників

з'являється можливість оптимізувати процес, щоб зробити час програмної розробки максимально малим та передбачуваним.

На практиці Канбан, зазвичай, включає наступні кроки [40]:

- Створення мапи потоку, пошук задач для розробки;
- Створення списку вимог замовника через набір задач для розробки;
- Залежно від видів діяльності та складу команди, вимірюється максимальний ліміт задач на один часовий сегмент розробки;
- Розроблення політики для вибору задач розробниками та вирішення питань, пов'язаних із потоком розробки;
- Розробка графіку та типового формату зустрічей.

Метод розвитку динамічних систем (DSDM) – у цієї технології чітко визначені фази, підфази, ролі та принципи, що дозволяють команді розробки програмного забезпечення ефективно працювати [98].

DSDM має три основні фази: передпроєктну, проєктний цикл та постпроєктну фазу. З рис. 1.21 бачимо, що життєвий цикл DSDM є ітераційним. Це означає, що незважаючи на існування чіткої послідовності фаз, проєкт може розвиватися ітераційно. В фазі життєвого циклу проводиться дослідження техніко-економічного обґрунтування, і після завершення бізнес-кейсу компанія визначає функціональні та нефункціональні вимоги високого рівня та розробляє архітектуру програмного забезпечення та план його розвитку та впровадження. Після цього детально описується оцінка витрат та вигод на високому рівні. На другому етапі в фазі життєвого циклу керівники проєкту та команда з розробки проєкту тісно співпрацюють із замовником, щоб розробити функціональну модель. Процес розробки розпочнеться на третій фазі. Проєкт розробляється в ітераціях, і замовник постійно бере участь в цьому процесі, щоб переконатися, що програмне забезпечення відповідає його бізнес-потребам. На останньому етапі готову систему передають замовнику та очікують на зворотній зв'язок.

Було проведено аналіз на виявлення загальних рис в продемонстрованих Agile технологіях. В цілому, виявлені загальні риси можна розділити на дві

категорії: переваги та недоліки Agile процесу, які притаманні усім Agile технологіям.

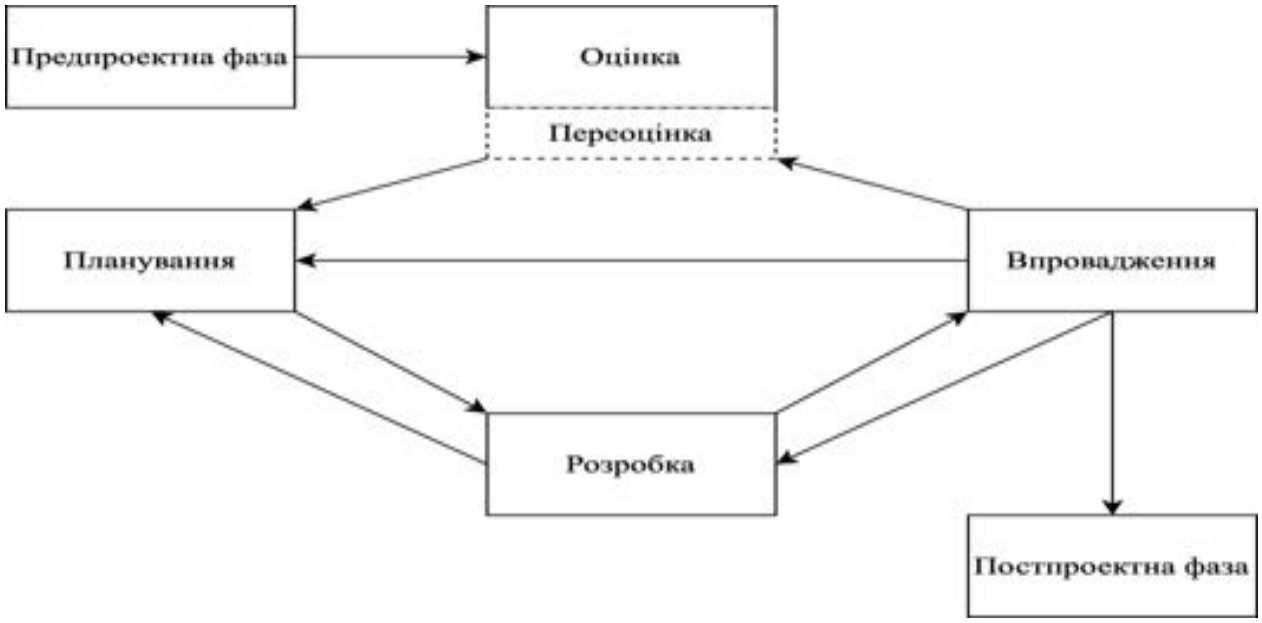


Рисунок 1.21 – Життєвий цикл DSDM (Джерело – [98])

Переваги Agile процесу:

- *Адаптивність до мінливого середовища:* з меншими циклами планування легко прийняти зміни в будь-який час проєкту. Завжди є можливість вдосконалити та переосмислити товарний запас продукту, дозволяючи командам впроваджувати зміни до проєкту в рамках однієї або декількох ітерацій.
- *Кінцева мета може бути невідома:* Agile дуже корисний для проєктів, де кінцева мета чітко не визначена. По мірі прогресу проєкту цілі стають більш чіткими і розвиток може легко адаптуватися до пов'язаних з цим змін.
- *Швидкість і якість:* розбиття проєкту на ітерації дозволяє команді зосередитись на високоякісній розробці та тестуванні. Проведення тестування під час кожної ітерації означає, що помилки ідентифікуються та вирішуються раніше, що призводить до більш швидкого впровадження високоякісного програмного забезпечення.

- *Інтегрованість з замовником:* у замовників є багато можливостей побачити прогрес роботи та реально вплинути на кінцевий продукт. Вони можуть здобути почуття власності, працюючи так тісно з командою проєкту. Також Agile проєкти заохочують відгуки користувачів та членів команди протягом усього проєкту, тому засвоєні уроки використовуються для покращення майбутніх ітерацій.

Недоліки Agile процесу:

- *Кінцевий продукт може відрізнятися від запланованого:* з початку проєкт може не мати остаточного плану, тому кінцевий продукт може виглядати інакше. Це випливає з того, що оскільки Agile є досить гнучким і нові ітерації можуть бути додані на основі зворотного зв’язку з замовником і це, в свою чергу, може призвести до зовсім інших результатів.
- *Документацією можна знехтувати:* Agile надає перевагу робочому програмному забезпеченню перед вичерпною документацією, тому деякі члени команди можуть відчути, що менш важливо зосередитись на документації. Хоча вичерпна документація сама по собі не призводить до успіху проєкту, Agile команди повинні знайти правильний баланс між документацією та розробкою.
- *Більш корисний для менеджменту, ніж для розробки:* Agile технологія допомагає керівництву приймати рішення щодо розробки програмного забезпечення, встановлювати цілі для розробників та встановлювати термін для цих цілей. Але розробникам може бути важко впоратись із постійно мінливим середовищем і щоразу змінюючи вже розроблену функціональність.
- *Взаємодія з клієнтами є ключовим фактором розробки успішного програмного забезпечення:* Agile процес базується на залученні до процесу розробки замовника, оскільки весь проєкт розробляється відповідно до вимог, що ним задаються. Отже, якщо замовнику не зрозуміло особливості продукту, процес розробки скоріше за все вийде за попередньо сплановані часові та бюджетні рамки.

Продемонструємо загальне порівняння Agile технологій [84] – табл. 1.5. Можна зробити висновок, що Scrum технологія є найпопулярнішою в середовищі розробки аутсорс проєктів.

Розглянемо цю технологію детальніше. Як вже згадувалося раніше, це ітеративна модель розробки програмного забезпечення. Ітерації з фіксованою довжиною, що називаються спринтами, дозволяють команді регулярно постачати програмне забезпечення. Наприкінці кожного спринту збираються зацікавлені сторони та члени команди, щоб спланувати наступні дії.

Scrum дотримується набору ролей, обов'язків та зустрічей, які ніколи не змінюються. Наприклад, Scrum закликає чотири церемонії, що забезпечують структуру кожного спринту: планування спринту, щоденну зустріч для синхронізації процесу, демонстрацію спринту та ретроспективу спринту.

Основні переваги:

- *Прозорість процесу:* завдяки щоденним зустрічам, вся команда знає, хто чим займається, усуваючи багато непорозумінь та плутанини. Проблеми визначаються заздалегідь, що дозволяє команді вирішити їх, перш ніж вони вийдуть з-під контролю;
- *Відповідальність команди:* команда колективно вирішує, яку роботу вони можуть виконати у кожному спринті. Всі вони працюють разом, удосконалюючи процес співпраці та разом з цим, надаючи можливість кожному члену команди працювати незалежно;
- *Пристосування до зміни:* з короткими спринтами та постійним зворотним зв'язком легше впоратися з незапланованими змінами;
- *Економія витрат:* постійне спілкування робить команду обізнаною про всі проблеми та зміни, як тільки вони виникають, допомагаючи знизити витрати та підвищити якість. Завдяки кодуванню та тестуванню функцій на кожній ітерації, існує постійний зворотний зв'язок і помилки можна виправити на початку, перш ніж вони стають занадто дорогими для виправлення.

Таблиця 1.5 – Порівняння Agile технологій (Джерело – [84])

Характеристики	**XP**	**Scrum**	**DSDM**	**FDD**
1	2	3	4	5
Підхід	Ітеративно інкрементальний	Ітеративно інкрементальний	Ітеративний	Інкрементальний
Період часу для ітерації	1-6 тижнів	2-4 тижні	80% рішення за 20% часу	Від 2 днів до 2 тижнів
Розмір команди	Маленькі команди (< 20 членів)	Будь-які	Незалежні команди всіх розмірів	Багато членів і більше однієї команди
Слушний розмір проєкту	Для менших проєктів	Усі типи проєктів	Усі типи проєктів	Більш складні проєкти
Залучення користувачів	Користувач сильно задіяний	Залучення через власника продукту	Залучення через часті релізи	Залучення через часті звіти
Документація	Лише основна документація	Лише основна документація	Документація існує	Документація важлива
Основні практики	Запроси користувачів, розробка через тестування, рефакторинг, парне програмування	Спринт, товарний запас продукту та спринта, скрам мітинги	Прототипування, техніко-економічне обґрунтування та вивчення бізнесу	UML-діаграми

Основні недоліки:

- *Ризик розширення спринту:* деякі проєкти можуть зазнати розширення спринту через відсутність конкретної кінцевої дати. Без дати завершення, зацікавлені сторони можуть спокуситись продовжувати запитувати додаткову функціональність;
- *Некваліфікований Scrum-майстер може розвалити проєкт:* на відміну від менеджера проєкту, Scrum-майстер не має повноважень для керування командою. Якщо Scrum-майстер намагатиметься контролювати команду, проєкт не вдасться.
- *Погано визначені завдання можуть призвести до неточностей:* заздалегідь оцінені витрати та терміни проєкту не будуть точними, якщо задачі не визначені правильно. Якщо початкові цілі незрозумілі, планування стає складним і спринти можуть зайняти більше часу.

Для більш детальнішого розуміння процесу розглянемо основні ролі в процесі.

Власник продукту. Має бачення того, що він хоче побудувати, і передає це бачення команді. Власник продукту зосереджується на вимогах бізнесу та ринку, визначаючи пріоритетність роботу, яку необхідно виконати. Створює та керує товарним запасом продукту і взаємодіє з командою та іншими зацікавленими сторонами, щоб переконатися, що всі розуміють задачі у товарному запасі продукту. Власник продукту не є менеджером проєкту. Замість того, щоб керувати статусом та прогресом, його завдання – надавати інформацію команді щодо мети та баченням;

Scrum-майстер. Організовує зустрічі, вирішує проблеми, виникаючі в процесі управління проєктом та співпрацює з власником продукту, щоб забезпечити готовність продукту до наступного спринту. Він також гарантує, що команда слідкує за процесом. Він не має повноважень щодо управління членами команди, але він має повноваження щодо управління процесом;

Команда. Зазвичай складається від п'яти до семи членів. Усі в проєкті працюють разом, допомагають один одному. Команда знає план для кожного

спринту і передбачають, скільки роботи вони можуть виконати за кожну ітерацію.

І нарешті розглянемо етапи Scrum процесу для розробки аутсорс проєктів [118]:

1. *Товарний запас продукту.* Власник продукту та команда збираються для розстановки пріоритетів в задачах, що містяться в товарному запасі продукту. Треба розуміти, що товарний запас продукту – це не список речей, які слід завершити, а скоріше список усіх бажаних функцій продукту. Команда розробників вибирає роботу з товарного запасу продукту для наповнення функціональності кожного спринту.
2. *Планування спринту.* Перед кожним спринтом власник продукту представляє команді головні елементи з товарного запасу продукту на зустрічі з планування спринту. Потім команда обирає, яку роботу вони можуть виконати під час спринту, і переміщує роботу з товарного запасу продукту на майбутній спринт.
3. *Щоденні Scrum зустрічі.* На цих зустрічах кожен член команди розповідає про свої цілі та будь-які проблеми, які виникли.
4. *Огляд спринту.* Наприкінці кожного спринту команда представляє роботу, яку вони виконали. Ця зустріч повинна містити демонстрацію щодо виконаної функціональності.
5. *Ретроспектива спринту.* Також наприкінці кожного спринту команда розмірковує про те, наскільки добре була виконана робота в цьому спринті, і планує про будь-які зміни, які необхідно ввести в наступному спринті для підвищення ефективності роботи команди.

Беручи до уваги надану інформацію, не дивно, що Scrum технологія набула більшої популярності у процесі розробки аусорс проєктів, ніж інші Agile технології. Здебільшого, така ситуація склалася через досить сильну залученість замовника у процесі розробки проєкту та відсутності необхідності у постійному контролі процесу розробки, що дозволяє делегувати процес розробки майже повністю аутсорс компанії. Цей процес дуже гнучкий, адже

дозволяє динамічно змінювати функціональність продукту без сильного впливу на попередньо зроблені оцінки тривалості та бюджету. Також гнучкість досягається за рахунок того, що у найкращому випадку замовник залучає до процесу тільки одну людину – власника проєкту, що у свою чергу, зменшує витрати на місцевий штат.

Висновки до розділу 1

Поточна економічна ситуація та стан IT-індустрії України і світу створили умови, в яких є потреба ретельного аналізу процесу управління аутсорсинговими IT-проєктами з метою підвищення його ефективності на всіх етапах. Ця ситуація виникає внаслідок зростаючої кількості проєктів, високої поточної та потенційної конкуренції з боку IT-індустрій інших країн, а також за рахунок впливу зовнішньо-економічних факторів: пандемія COVID-19, Російсько-Українська війна, тощо. Таким чином, набуває актуальності розробка та впровадження нестандартних підходів, які поєднують сучасні технології, що властиві IT-галузі, з методами, що враховують різноманітні зміни у середовищі реалізації як окремих проєктів, так і комплексних портфоліо. Залучення методів нечіткої математики може бути одним з перспективних напрямків покращання управління.

Використання апарату нечіткої логіки може сприяти підвищенню ефективності управління ризиками та невизначеністю, притаманних портфелям проєктів. Нечітка логіка дозволяє враховувати непевність і неоднозначність в процесі прийняття рішень, що особливо важливо в умовах високих фінансових та соціальних ставок, так як допомагає вирішити проблеми, які не можуть бути враховані стохастично. Класичний стохастичний підхід менш придатний у зв'язку зі складністю і нестабільністю фінансових ринків. Таким чином,

використання апарату нечіткої логіки допомагає зменшити невизначеність і підвищити ефективність процесів управління портфелем проєктів.

Аналізуючи наявну інформацію, можна зробити висновок, що використання нечіткої логіки для вирішення задач управління ІТ-проєктами є відносно нерозповсюдженим та потребує подальшого розвитку. Наразі більша увага приділяється загальному процесу управління проєктами: будуються різноманітні фреймворки, які спрямовані на його описання та подрібнення на окремі задачі. Значно менша увага приділяється процедурам розв’язання конкретних завдань у процесі управління ІТ-проєктами. Досить рідко можна зустріти поодинокі дослідження щодо окремих підзадач процесу з використанням нечіткої логіки та ще більш унікальним є використання нечіткої логіки для процесу загалом.

Таким чином, з метою уникнення цих обмежень та враховуючи потребу в покращенні процесу управління ІТ-проєктами, було прийнято рішення щодо реалізації типової оболонки, яка базується на системах нечіткої логіки, для вирішення задач управління ІТ-проєктами. Цей підхід дозволить враховувати невизначеність та неповноту інформації, яка виникає в процесі управління ІТ-проєктами, та забезпечить більш точне та ефективне прийняття рішень.

Відповідно до мети роботи та аналізу літературних джерел було виявлено невирішені завдання в процесі управління ІТ-проєктами:

1. Створення моделей прийняття рішень із залученням методів, які враховують важливі економічні аспекти та відносяться до групи нечіткої логіки. Застосування таких моделей допоможе враховувати вартість проєктів, рентабельність, оптимізацію витрат, оцінку ефективності та інші економічні фактори в ході розробки й обгрунтування рішень.
2. Розв'язання проблем, які виникають в умовах неповноти і невизначеності інформації та проєктних характеристик. Управління ІТ-проєктами залежить від точності та достовірності доступної інформації, але у багатьох випадках інформація є неповною або недостовірною. Використання нечіткої логіки дозволить ефективно враховувати

невизначеність у процесі прийняття рішень, що сприятиме покращенню якості та результативності управління ІТ-проєктами.

3. Сучасні дослідження процесу управління портфоліо проєктів з використанням нечіткої логіки у більшості розглядають підпроцеси вибору проєктів та передбачення ризиків їх реалізації, хоча спектр проблем, які можна вирішити, використовуючи нечітку логіку, значно більший. Наразі більшість досліджень у цій області фокусуються на підпроцесах вибору проєктів та передбаченні ризиків, але широкий спектр проблем, які можна вирішити, використовуючи даний математичний апарат, залишається недостатньо дослідженим. Розширення застосування нечіткої логіки у процесі управління портфелем проєктів дозволить ефективно управляти різноманітністю проєктів, оцінювати їх пріоритетність, враховувати ризики та ресурсні обмеження.

Зазначені чинники обумовлюють потребу та актуальність використання нечіткої логіки у процесі управління ІТ-проєктами, а саме − актуальність розв'язання наукового завдання. Подальші дослідження та розробки в цій галузі сприятимуть вдосконаленню методів та інструментів управління, що приведе до поліпшення результатів і ефективності реалізації ІТ-проєктів.

Результати розділу 1 висвітлено у наступних працях автора, наведених у Додатку А: [3, 5, 6, 9].

РОЗДІЛ 2
ІНСТРУМЕНТАЛЬНО-АНАЛІТИЧНЕ ЗАБЕЗПЕЧЕННЯ УПРАВЛІННЯ АУТСОРСИНГОВИМИ ІТ-ПРОЄКТАМИ: АПАРАТ НЕЧІТКОЇ ЛОГІКИ В КОНТУРІ УПРАВЛІННЯ ПОРТФОЛІО ПРОЄКТІВ

2.1 Робота з нечіткими знаннями як основа формування інструментально-аналітичного забезпечення управління аутсорсинговими ІТ-проєктами

Розв’язання проблеми невизначеності у контексті нечіткого підходу до управління портфелем проєктів привертає увагу багатьох дослідників, що знайшло відображення у певному колі літературних джерел [24, 47, 70, 78].

Вище було доведено, що невизначеність має подвійне значення. Згідно з цим, обговорюючи управління портфелем, розглядаються обидва значення.

По-перше, робиться акцент на необхідності визначення способів виміру та компенсації неточності. Можливим інструментом для завдання є системи нечіткої логіки, тобто використання нечітких чисел та нечітких множин для опису невизначених явищ та / або використання нечіткої логіки для обробки даних з невизначених явищ. Базою комплексного підходу до управління портфелем проєктів в умовах невизначеності є повне врахування нечітких підфаз, а саме:

- Нечітка вхідна інформація – приведення даних з портфоліо та навколишнього середовища до стану, прийнятного системами нечіткої логіки.
- Обробка нечіткої інформації в основному використовує нечітку логіку та нечітку математику. Є вже багато пропозицій такого типу для оцінки значущих змінних та створення рішень. Деякі з них навіть пропонують способи нечіткого відбору та оцінки рішень нечіткими функціями.

- Випуск нечіткої інформації буде етапом для приведення інформації в прийнятний для використання вигляд, що поведе за собою прийняття рішення та здійснення управлінських дій на портфоліо.

Завдяки тому, що нечіткий підхід до вирішення проблем в умовах невизначеності стає все більш популярним серед дослідників, цілком ймовірно, що існує вже широкий спектр запропонованих рішень для різних фаз та / або завдань процесу управління проєктами. Доречно розглянути актуальність використання нечіткої логіки для управління ІТ-проєктами.

Для управління процесами програмного забезпечення необхідно спершу знати, як їх виміряти – з такою основою можна робити прогнози, генерувати плани, відслідковувати діяльність, контролювати розумно та аргументовано, і нарешті оцінювати процеси та дізнаватися, як їх можна вдосконалити. Класифікації та прогнози, що складають частину цієї управлінської діяльності, можуть бути побудовані з формальних моделей, експертних знань або якоїсь відповідної комбінації.

Використання таких програмних моделей вимірювання для прогнозування та класифікації в управлінні проєктами обіцяло багато, а в деяких випадках було дуже успішним. Однак з ними також пов’язано багато труднощів, не останнє з яких полягає у забезпеченні того, щоб моделі насправді фіксували фактори, що цікавлять та дійсно впливають на процес управління. Менеджери можуть володіти значними знаннями про ймовірні зв’язки між факторами, включаючи змінні, які неможливо ефективно чи легко моделювати у формальному чи кількісному відношенні. Ці змінні досвідчений менеджер неявно враховує, коли робить "здогадку" або інстинктивно округляє передбачуване значення вгору або вниз після використання формальної методики. Однак для покращення процесів потрібно мати повторюваність та послідовність управління. Це, як правило, вимагає використання більш формалізованих методів моделювання, які менш здатні враховувати такі суб'єктивні знання.

Зважаючи на наведені труднощі, одним з найбільш часто використовуваних "прийомів прогнозування" в практиці управління, особливо в нових програмних організаціях, є експертна оцінка [57]. Хоча така оцінка може бути дуже ефективною, вона страждає від проблем суб'єктивності, неповторності, непослідовності та вразливості до втрати знань, якщо менеджери покинуть організацію. У цих умовах потрібні не більш формалізовані моделі метрик та не більше способів відгадування значень параметрів – скоріше потрібно знайти спосіб поєднання експертного та формального моделювання, не надто применшуючи переваги будь-якого з них. Таким чином найважливішими факторами на користь використання нечіткої логіки є:

Здатність впоратись із невизначеністю: терміни «прогнозування», «оцінка» та інші зазвичай розуміють як прогнозне значення. З точки зору бізнесу в наших інтересах є надання оцінок, максимально наближених до фактично досягнутих. При цьому слід також визнати, що існує певний шанс, що фактична величина не буде відповідати зробленій оцінці. Такий результат стає все більш імовірним, коли є лише мінімальна інформація, на якій можна базувати своє передбачення. У цьому випадку оцінка може бути зроблена більш точною в ході процесу, коли буде доступно більше інформації;

Варіативність у деталізації: наступною перевагою нечітких логічних моделей є можливість змінювати рівень деталізації на входах і виходах без зміни базового набору правил [54]. Під час аналізу вимог вхідні та вихідні дані моделі є лінгвістичними, вони стають нечіткими числами під час проєктування та перетворюються на точні значення на більш пізніх етапах, коли відомо більше даних;

Здатність впоратися з мінімальними даними: оскільки нечіткі моделі можуть базуватися на даних, знаннях або тому і іншому разом, ці моделі можуть бути розроблені з невеликою кількістю вхідних даних. Це значна перевага, враховуючи широко визнані проблеми, що виникають при зборі даних в дослідженнях та практиці управління програмним забезпеченням.

Добре відомо, що збір однорідних наборів даних часто ускладнюється швидко мінливими технологіями та небажанням міжорганізаційного обміну інформацією. Навіть у межах однієї організації може бути чималий тиск з боку програмістів та менеджерів проти збору даних;

Стійкість до характеристик набору даних: нерідко буває що набори даних управління програмним забезпеченням містять незвичайні або аномальні спостереження, таким чином знижуючи загальну точність будь-якої моделі, отриманої з цих даних емпірично [75]. Ці нетипові спостереження можуть виникати з різних причин – зміни в практиці розробки, посилене навчання персоналу або інші незмірні впливи. Розробляючи моделі із значним залученням експертів, де модель може бути інтерпретована, перевірена та доопрацьована за потреби, деякі проблеми нерепрезентативних даних можна зменшити або уникнути;

Використання експертних знань: оскільки нечітка логіка дозволяє представляти поняття як функції членства та асоціації між цими поняттями як правилами, можливо дуже легко включати експертні знання про такі поняття та відносини у нечіткій системі. Таким чином, ці експертні знання природно зафіксовані в системі, забезпечуючи спосіб їх збереження поза роботою експерта;

Легко вивчаємє та більш прозоре: відносно кажучи, логічне моделювання значно легше зрозуміти та використовувати, ніж більшість методів статистичної та нейронної мережі. Незважаючи на те, що існує кілька важливих математичних принципів, техніка піддається як початківцям, так і експертам. Нечіткі логічні моделі також є повністю зрозумілими, оскільки перехід від вхідних даних до вихідних може бути проаналізовано та розглянуто, де це необхідно. Ідеальний метод моделювання повинен забезпечити баланс між ефективністю з точки зору точності та послідовності, з одного боку, та простотою у розрахунку та застосуванні з іншого.

Використання нечіткої логіки надає низку переваг [5]:

- Простота та гнучкість опису правил, за якими нечітка система приймає рішення.
- Можливість описати правила у псевдо-лексичній формі, що спрощує їх розуміння.
- Можливість приймати рішення при наявності неповної інформації.
- Простіші у розробці, адже для такого роду систем не є обов'язковою детермінованість.

Однак використання нечіткої логіки накладає і деякі обмеження на систему. В рамках конкретної задачі, яку планується вирішувати за допомогою нечіткої логіки, є необхідність слідкувати за станом бази знань та бази даних. Дуже важливо тримати їх у відповідності один до одного, щоб система мала змогу робити коректні рішення.

Формалізація знань пов’язана з проблемою ускладнення використання традиційного математичного апарату. Це пояснюється необхідністю визначення понять, що оперують якісними характеристиками досліджуваних об’єктів. Такі характеристики не можуть бути однозначно інтерпретовані або є «розмитими». Окрім цього, інтелектуальні системи часто стикаються з необхідністю використання неточних знань.

Принципи роботи та процедури обробки неточних знань детально розглянуто рядом авторів у низці робіт. Наприклад, в [11] розглянуті види неточних знань, класифікація невизначеностей та основні підходи щодо їх обробки, включаючи теорію функцій довіри та нечітку логіку.

Нечітка логіка або теорія нечітких множин була запропонована американським математиком Лотфі Заде у 1964 році. Потім Бартоломео Коско була досліджена взаємозалежність нечіткої логіки і теорії нейронних мереж та доведена, так звана, FAT-теорема (Fuzzy Approximation Theorem), яка підтвердила повноту нечіткої логіки.

В роботах Марії Земанкової та інших вчених були закладені основи теорії нечітких систем управління базами даних, основними властивостями яких є оперування неточними даними, обробка нечітко заданих питань, а також

використання якісних параметрів поряд з кількісними. Була розроблена нечітка алгебра – наука, що дозволяє при обчисленнях використовувати як точні, так і приблизні значення перемінних. Останні розробки в цій галузі – нечіткі когнітивні моделі Коско (Fuzz Cognitive Maps), на яких базується більшість сучасних систем динамічного моделювання в фінансах, політиці та бізнесі.

Сьогодні нечітка логіка стала однією з найбільш успішних сучасних технологій при розробці складних систем аналізу і управління.

Таким чином, нечітка логіка або нечітка математика (у своєму сучасному, розвинутому вигляді) є одним з найбільш ефективних підходів до розв'язання різноманітних проблем, яка враховує і невизначеність, і нечіткість, притаманні економічним дослідженням.

Дійсно, задача прийняття рішень в процесі управління проєктами в умовах невизначеності та мінливого середовища може бути охарактеризована наступним кортежем [11,с. 11]:

$$\langle A, X, C, f, P_s, D, T \rangle \ ,$$

де **A** – множина альтернатив;

X – множина висновків альтернатив;

C – векторний критерій оцінки висновків;

f – відображення множини X у множину векторних оцінок;

P_s – структура переваг особи, що приймає рішення (ОПР);

D – вирішальне правило або алгоритм, що дозволяють робити необхідні дії T над множиною альтернатив A.

Таким чином, X, K, f, P_s, D складають середовище і систему переваг ОПР, що виражається сукупністю множин критеріїв, альтернатив та висновків з відносинами переваги.

У нечіткому середовищі нечітко (у виді нечітких відносин і понять) можуть бути виражені всі елементи кортежу. Іншими словами, експертні оцінки альтернативних варіантів за критеріями можуть бути представлені як нечіткі множини, визначені за допомогою функцій належності.

Основи нечіткої логіки представлені у ряді літературних джерел [8-18, 20, 22, 50, 59, 125, 126].

Узагальнюючи схематично викладені засади обробки неточної інформації, на рис. 2.1.1 [66, с. 33] наведемо схему співвідношень класичних, експертних та нечітко-множинних ймовірнісних описів.

Дійсно, з рис. 2.1 видно, що з посиленням невизначеності класичні ймовірнісні описи поступаються, з одного боку, суб'єктивним ймовірностям, що засновані на експертній оцінці, а з другого боку, ймовірностям, що визначені якісно (приблизно), а не кількісно.

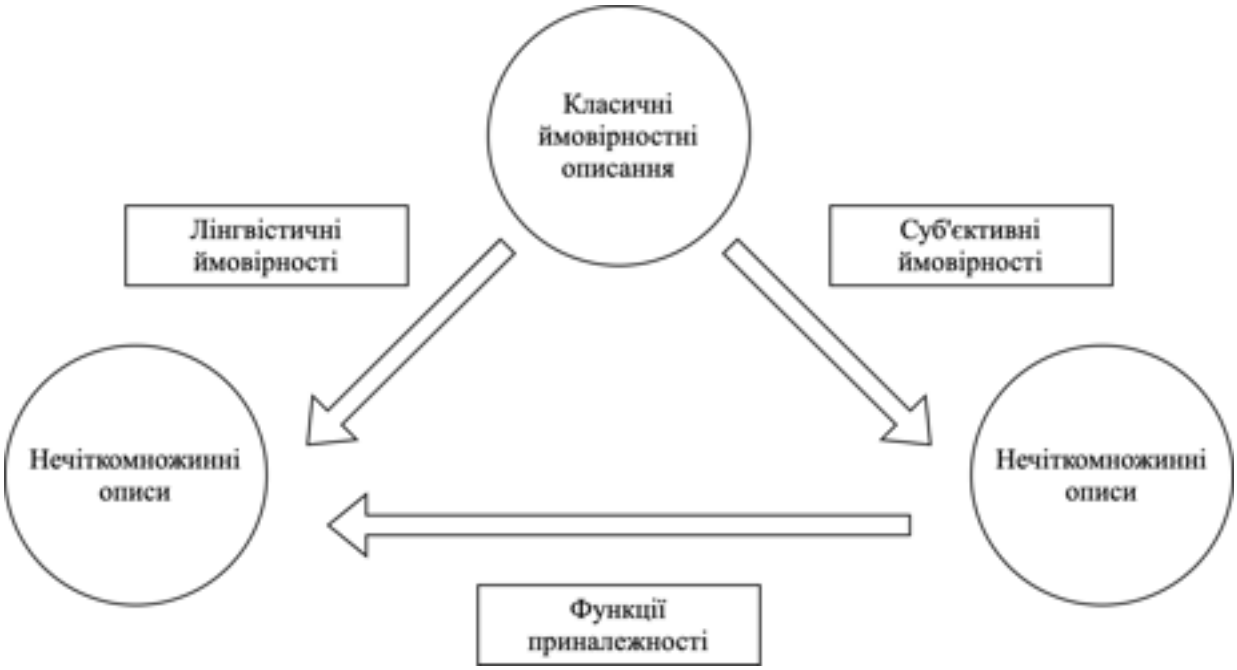


Рисунок 2.1 – Співвідношення класичних, експертних та нечітко-множинних стохастичних описів (Джерело − [66, с. 33])

Відповідно до цього точечні оцінки стохастичних розміщень (розподілів) заміщуються інтервальними (для експертних методів) та нечіткими (для методів нечіткої математики).

Таким чином, головні переваги саме нечітких інтелектуальних систем полягають в наступному:

− Можливість оперувати вхідними даними, заданими нечітко: наприклад, значення, що безперервно змінюються в часі (динамічні

задачі); значення, які неможливо задати однозначно (результати статистичних досліджень, рекламної кампанії тощо).

- Можливість нечіткої формалізації критеріїв оцінки та порівняння: оперування критеріями “більшість”, “можливо”, “переважно” та т. і.
- Проведення якісного аналізу як вхідних даних, так і кінцевих результатів (мається на увазі не стільки оперування саме значеннями даних, скільки ступенем їхньої достовірності і розподілом).
- Можливість проведення моделювання складних динамічних систем в оперативному режимі та порівняльного аналізу отриманих результатів із наданим ступенем точності. Це стає можливим завдяки тому, що оперування принципами поведінки системи (описаними нечіткими методами) не потребує багато часу: не витрачається час на визначення точних значень змінних та рівнянь, за якими вони обчислюються. Крім того, є можливість оцінки різних варіантів вихідних значень.

Як висновок, варто відмітити, що розвиток нечіткої математики та основні переваги нечітких інтелектуальних систем призвели до розвитку апарату експетних систем в цілому та, зокрема, апарату нечітких експертних систем.

2.2 Концептуальна платформа розробки нечіткої експертної системи FuzzyKIDE: архітектура, модельна основа та механізм логічного виводу

2.2.1 Архітектура типової експертної системи

Процеси створення та використання експертних систем в різноманітних предметних галузях належать до одного з напрямків штучного інтелекту.

Практичні успіхи, досягнуті у тій сфері, підтверджують досягнення зрілості наведеної області досліджень [6, 81, 95, 117].

Ядром будь-якої експертної системи є сукупність знань про конкретні явища та процеси – база знань. Тому не випадково технологія створення експертних систем (ЕС) зветься інженерією знань. До найбільш розповсюджених визначень експертних систем належать наступні.

Експертні системи (ЕС) − це клас систем зі штучного інтелекту, які призначені для отримання, накопичення, коригування знань з конкретної предметної галузі (запропонованих, як правило, експертами), виводу нових знань, рішення на основі отриманих знань практичних задач із поясненням процесу знаходження рішення [19, с. 11].

Згідно [21] експертна система визначається, як комп'ютерна програма, яка використовує експертні знання для високоефективного рішення задач в деякій вузький предметній галузі. Такі програми, як правило, представляють знання символічно, досліджують та пояснюють свої процеси міркувань та призначені для тих предметних галузей, в яких людині для досягнення майстерності необхідні роки спеціального навчання та практики.

ЕС мають такі основні властивості:

- алгоритм рішення є невідомим заздалегідь, тобто будується самою системою за допомогою символьних міркувань, які базуються на евристичних засобах;
- «прозорість» процесу отримання рішення − система усвідомлює, як вона отримала рішення;
- можливості щодо придбання нових знань, які надходять від користувача-експерта, а також проведення відповідних змін у поведінці згідно з отриманими знаннями;
- наявність метакомпоненти, тобто можливість проведення аналізу та пояснення (на його підставі) власних знань та конкретних дій;
- можливість забезпечення гнучкого інтерфейсу з користувачем (на природній мові).

Таким чином, експертні системи наділені можливостями експертів-фахівців в конкретній галузі. Вони націлені на отримання наступних результатів:

- автоматизація складних комплексних ітеративних процесів;
- підвищення продуктивності праці (підвищується ефективність та обсяг корисної інформації) та відповідне скорочення часу для досягнення конкретної мети у рішенні задач.

Базою експертних систем є дедуктивні міркування. Системи демонструють ефективні здібності в рішенні неструктурованих та погано визначених задач (це робиться шляхом залучення евристик, що доцільно у ситуаціях, коли недостатність необхідних знань та часу виключає можливість проведення повного аналізу). Але все це стосується конкретних предметних галузей (на відміну від інших програмних продуктів, що використовують процедурний аналіз).

Загальна архітектура типової експертної системи наведена на рис. 2.2. Загальними складовими ЕС є наступні:

- база знань;
- база даних;
- модуль логічного виводу (або «решатель», або «машина виводу»);
- модуль пояснень;
- інтелектуальний редактор бази знань;
- діалоговий компонент або інтерфейс з користувачами та експертами.

База даних призначена для зберігання вхідних та проміжних даних, які стосуються задачі, що вирішується у поточний час. Цей термін не треба використовувати як синонім стосовно систем управління базами даних та інформаційно-пошукових систем, де він застосовується для визначення довгострокової інформації. Бази даних експертних систем створюються тоді, коли обсяг вхідної інформації для використання в поточних розрахунках ЕС значний.

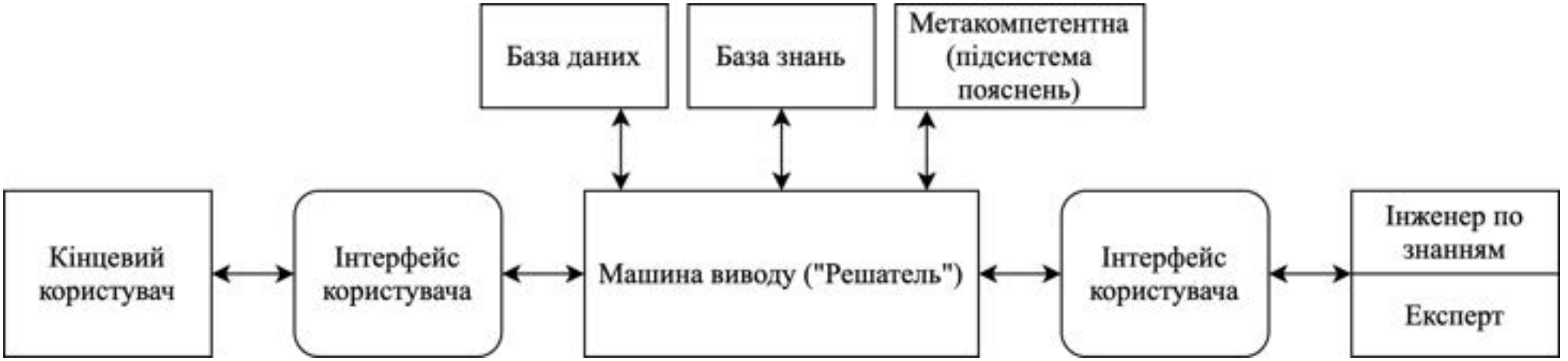


Рисунок 2.2 – Архітектура типової експертної системи

(Джерело – [8])

В загальному випадку бази даних можуть бути використані не тільки для розрахунків, але і для виконання логічних операцій. Тоді вони отримують статут файл-предикатів.

Включення бази даних до експертних систем націлене на розширення їх можливостей, тому що значно простіше включити в ЕС вже існуючу інформацію, ніж створювати нові файли даних. Це особливо актуально на сучасному етапі розвитку ЕС, коли розробники проєктують технології сумісного функціонування ЕС із існуючими комп'ютерними додатками.

Наприклад, найбільш революційний напрям в цієї галузі − сумісне використання ЕС та імітаційних моделей (імітаційних технологій).

База знань (БЗ) призначена для зберігання довгострокових даних, що описують галузь дослідження, та правил логічного виводу (з метою отримання висновків експертної системи).

Визначаються 2 складові бази знань − база фактів та база правил. Конкретне представлення знань в БЗ залежить від обраної форми та технології представлення.

Фактично, база знань містить відомі факти, які визначені у вигляді об'єктів, атрибутів та умов. Окрім описуючих ("описательных") представлень вона може містити висловлення невизначеності, тобто обмеження на ступінь достовірності факту.

База знань складається за допомогою знань експертів-консультантів − фахівців високого рівня в конкретній галузі.

Модуль логічного виводу використовує вхідні дані з бази даних та знання з бази знань та формує таку послідовність правил, яка за допомогою аналізу вхідних даних призводить до отримання рішення задачі.

Модуль логічного виводу виконує наступні задачі (функції):

- порівнює інформацію, яку містить запит користувача, із інформацією бази знань;
- знаходить конкретні цілі та причинні зв'язки;
- визначає порядок огляду та прикладення знань;

- додає в базу знань нові факти;
- оцінює відносну визначеність фактів, залучаючи до цього процесу відповідні коефіцієнти довіри, пов’язані з кожним фактом.

Модуль логічного виводу повинен функціонувати в умовах недостатності інформації.

Інтелектуальний редактор бази знань автоматизує процес наповнення експертної системи знаннями, який здійснюється користувачем-експертом. Його завданням є приведення знань у такий вигляд, щоб модуль логічного виводу міг використовувати їх у своїй роботі.

В найпростішому вигляді (тільки для внесення до бази знань) ці функції може виконувати текстовий процесор. Але, для сучасного рівня розвитку архітектури ЕС характерним є також виконання операцій перевірки даних (фактів та правил), що вводяться, на непротилежність з даними в базі знань.

Таким чином, інтелектуальний редактор бази знань містить систему ієрархічних меню, шаблонів мови представлення знань, підказок (режим help) та інших сервісних засобів, що полегшують роботу з базою знань.

Модуль пояснень призначено для отримання коментарів щодо кінцевих результатів виводу − чому система отримала саме таке рішення, які знання для цього використовувала. Модуль пояснень грає значну роль в технології створення та експлуатації ЕС, тому що полегшує експерту тестування системи та підвищує довіру користувача до отриманого результату.

Діалоговий компонент зорієнтовано на підтримку зв’язків зі всіма категоріями користувачів в ході проведення експертних консультацій із системою, а також для підтримки процесу придбання знань та пояснення результатів роботи.

Часто модуль придбання знань, модуль пояснень та діалоговий компонент відокремлюються у специфічний модуль, що зветься «діалоговий процесор» або «діалоговий монітор», який забезпечує взаємодію кінцевого користувача, а також інженера зі знань з експертною системою професійною, природною, обмеженою, графічною та т. і. мовами.

Засоби створення ЕС, зазвичай, відокремлюються. Це зроблено тому, що питання про різницю інструментальних засобів створення ЕС та програмного середовища саме ЕС є дискусійним. З одного боку, засоби створення ЕС, окрім мови, яка використовується для доступу до знань, а також для створення бази знань, містять засоби підтримки, що націлені на забезпечення взаємодії користувача з компонентою ЕС. У такому разі, до засобів підтримки належать діалоговий інтерфейс та компонента пояснень. І тоді тільки модуль логічного виводу є дійсно ЕС. З іншого боку, згідно з визначенням, засоби підтримки також є частиною закінченої ЕС.

Однак, на наш погляд, це питання не принципове, тому що не має суто прикладного значення. Під засобами створення ЕС нами розуміється мова програмування (що використовується з цією метою) та всі забезпечуючи засоби, окрім відокремленого діалогового інтерфейсу (лінгвістичний процесор та компонента пояснень).

2.2.2 Модель побудови бази знань FuzzyKIDE

Інтелектуальний процес людини надто складний, щоб його можна було представити як детермінований алгоритм. Однак більшість фахівців здатні сформувати свої знання у формі правил, використовуючи які виникає можливість вирішувати проблеми предметної області. Правило можна визначити як структуру ЯКЩО-ТО (IF-THEN), яка пов'язує інформацію або факти в частині IF з деякими діями / інформацією в THEN частині.

Правила відносно легко створити і зрозуміти. Ці правила часто називаються продукційними правилами, оскільки вони обов'язково призводять до певного результату при використанні. На основі цих правил створюються різні бази знань з механізмами прямого/зворотного виводу, що дозволяє знайти рішення доменних задач, зменшуючи загальну кількість обчислень.

На рис. 2.3 продемонстровано приклад класичного механізму продукційного виводу, на основі бази даних з необхідною для виводу інформацією, базою фактів для збереження проміжних результатів та бази знань з інформацією о способах виводу по базі знань та базі фактів.

Удосконалення процесу створення та роботи з базою знань на основі продукційних правил допомагає спростити технологію роботи з нею для фахівця; розширює спектр завдань, які з її допомогою можна вирішити, і підвищує точність кінцевого результату.

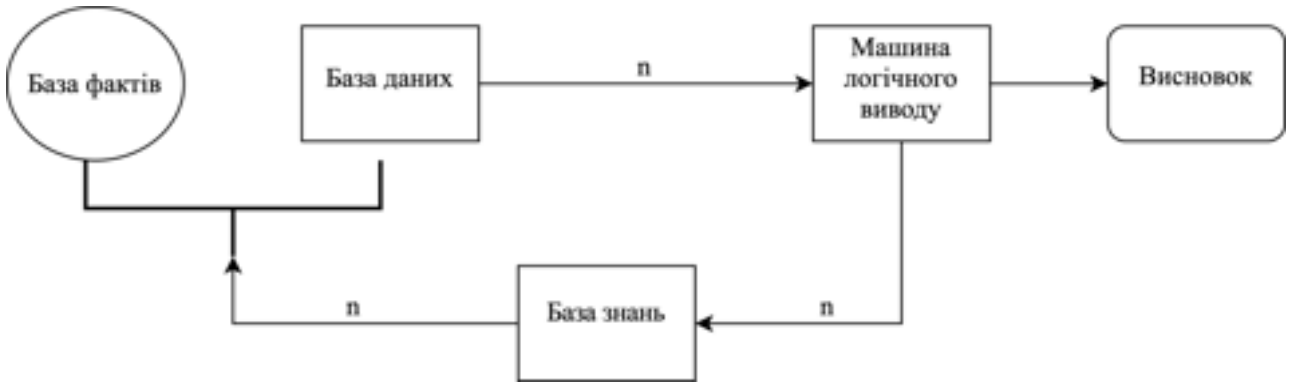


Рисунок 2.3 – Класичний механізм продукційного виводу

(Джерело – власна розробка)

Використання традиційних продукційних правил має ряд обмежень. Розглянемо наступний приклад традиційного продукційного правила:

$$IF\ (x > a)\ THEN\ (y = b) \qquad (2.1)$$

де $a, b \in \mathbb{R}$ та є чіткими значеннями.

Перший недолік цього правила полягає в тому, що воно використовує строгі порогові значення. Іншим недоліком цього правила є використання числових значень безпосередньо в тексті правила. У випадку складних правил, може бути досить важко дослідити і зрозуміти структуру бази знань, яка складається з таких правил.

Недоліком класичного механізму виводу, продемонстрованого на рис. 2.3 є постійна потреба в доступі до бази даних з метою отримання необхідної інформації для обчислення і підтримки бази фактів в поточному стані.

Недолік отримання інформації для обчислень досить істотний, але використання нечітких продукційних правил у порівнянні з класичними має позбутися цього дефекту. Другий недолік не настільки важливий, але він змушує замислитися про структуру і способи підтримки бази фактів, яка фізично може бути частиною бази знань, що є нелогічним або частиною бази даних, що збільшує навантаження на базу даних.

З метою усунення цих недоліків можна переписати правило (2.1) за допомогою нечітких продукційних правил:

$$IF\ (I_X\ IS\ IL_i)\ THEN\ (O_Y\ IS\ OL_j) \qquad (2.2)$$

У цьому варіанті опису правила вхідне значення I та вихідне значення O асоціюються з лінгвістичними змінними IL_i та IL_j за допомогою функцій належності до нечітких множин X та Y відповідно.

Через перехід від традиційних до нечітких продукційних правил вдалося позбутися недоліків цих правил. Але складність роботи з правилами зросла, тому що зараз існує потреба описати лінгвістичні змінні, які використовуються у цих правилах.

Пам'ятаємо, що лінгвістична змінна визначається наступним чином $< x, T, U, G, M >$, де x – назва змінної. Проаналізувавши поточну повну структуру лінгвістичних змінних, була розроблена оптимальна спрощена структура лінгвістичних змінних, яка може бути використана в базах знань. Продовжуючи приклад, опис лінгвістичних змінних можна представити таким чином:

$$\begin{aligned} I_X &= \{(x, \mu_I(x)) | x \in X\}, \quad X = [IL_1, IL_2, IL_3] \\ O_Y &= \{(y, \mu_O(y)) | y \in Y\}, \quad Y = [OL_1, OL_2, OL_3] \end{aligned}, \qquad (2.3)$$

де I_X і O_Y – імена лінгвістичних змінних,

X та Y – набори лінгвістичних значень для I_X і O_Y,

μ_I та μ_O – функції належності до нечітких множин X та Y.

Зазвичай лінгвістичні змінні визначаються користувачем, тому немає необхідності в процедурі генерування значень G для множини T. Опис

множин U і M об'єднуються в описі функцій належності до нечіткої множини.

Такий тип лінгвістичних змінних полегшує роботу користувача експертної системи, який використовує ці змінні [4]. Але ця структура, як і оригінал, не враховує джерело даних, з якого береться інформація для побудови функцій належності. Для цього попередню структуру (2.3) було змінено додатковою нотацією:

$$\begin{aligned} &I_X : Initial : \{(x, \mu_I(x)) | x \in X\}, \quad X = [IL_1, IL_2, IL_3] \\ &O_Y : Derivative : \{(y, \mu_O(y)) | y \in Y\}, \quad Y = [OL_1, OL_2, OL_3] \end{aligned}, \tag{2.4}$$

де *Initial* – позначає лінгвістичну змінну, значення якої буде взято, наприклад, з бази даних, а *Derivative* – позначає змінну, значення якої виводиться в процесі роботи з базою знань, тобто не вимагає наявності значень від бази даних.

2.2.3 Механізм логічного виводу FuzzyKIDE

Проблема ускладнення опису продукційних правил зменшується зі збільшенням кількості правил, з яких формується база знань, оскільки при збільшенні бази знань більш детальний її опис дозволяє користувачеві зрозуміти і працювати з нею досить швидко. Однак, зі збільшенням бази знань, тобто кількості правил у базі знань, виникають інші проблеми. Ці проблеми випливають з процесу роботи з цією базою знань, а саме з класичного механізму прямого виводу.

Можна позбутися обох недоліків, переглянувши спосіб роботи з базою знань та шляхи її підтримання. Поєднуючи описані в розділі 2.2.2 модифікації для лінгвістичних змінних і нечітких продукційних правил, була розроблена оптимальна структура бази знань. Ця структура бази знань поєднує дві частини: нечіткі продукційні правила і спрощені лінгвістичні змінні. Така структура бази

знань, продемонстрована на рис. 2.4, дозволяє позбутися бази проміжних фактів, яка була тісно пов'язана з необхідністю зберігати проміжну інформацію з метою постійного доступу до бази даних.

Водночас, це дозволяє використовувати семантичну мережу, яка забезпечує доступ до бази даних один раз для отримання початкових даних. При такому варіанті використання семантичної мережі реверсується залежність від початкових даних у базі даних таким чином, що ядру механізму виводу, тобто семантичній мережі, немає необхідності постійно робити запити за даними, а тільки розраховувати вже на наявні початкові дані та потенційну можливість їх розширення ззовні в процесі виводу.

Таким чином, це призводить до прискорення процесу нечіткого продукційного виводу.

Семантичні мережі були розроблені з метою представлення знань інтелектуальної системи, що використовує природну мову. Система обробки семантичної мережі SnePS [93] є позначеним спрямованим графом, в якому вузли представляють поняття, а дуги являють собою бінарні відносини між поняттями. Пропозиційні графіки з сімейства SNePS є графами, в яких кожен унікальний вираз з бази знань представлений вузлом у графі. Продукційне правило може бути представлено у графі через вузли самого правила, формул вхідного та вихідного аргументів, та дуг, які переходять від вузлів правила до вузлів аргументів. Також дуги позначають ролі, які аргумент відіграє в правилі, пояснюючи зв'язок. Також варто не забувати про зв'язок правил між собою через включення лінгвістичних змінних з правою чи лівою частини одного правила у праву чи ліву частину іншого. У разі, коли ліва частина одного правила зустрічається у правій частині другого, друге правило називається *попередником*. У протилежному випадку – *послідовником*.

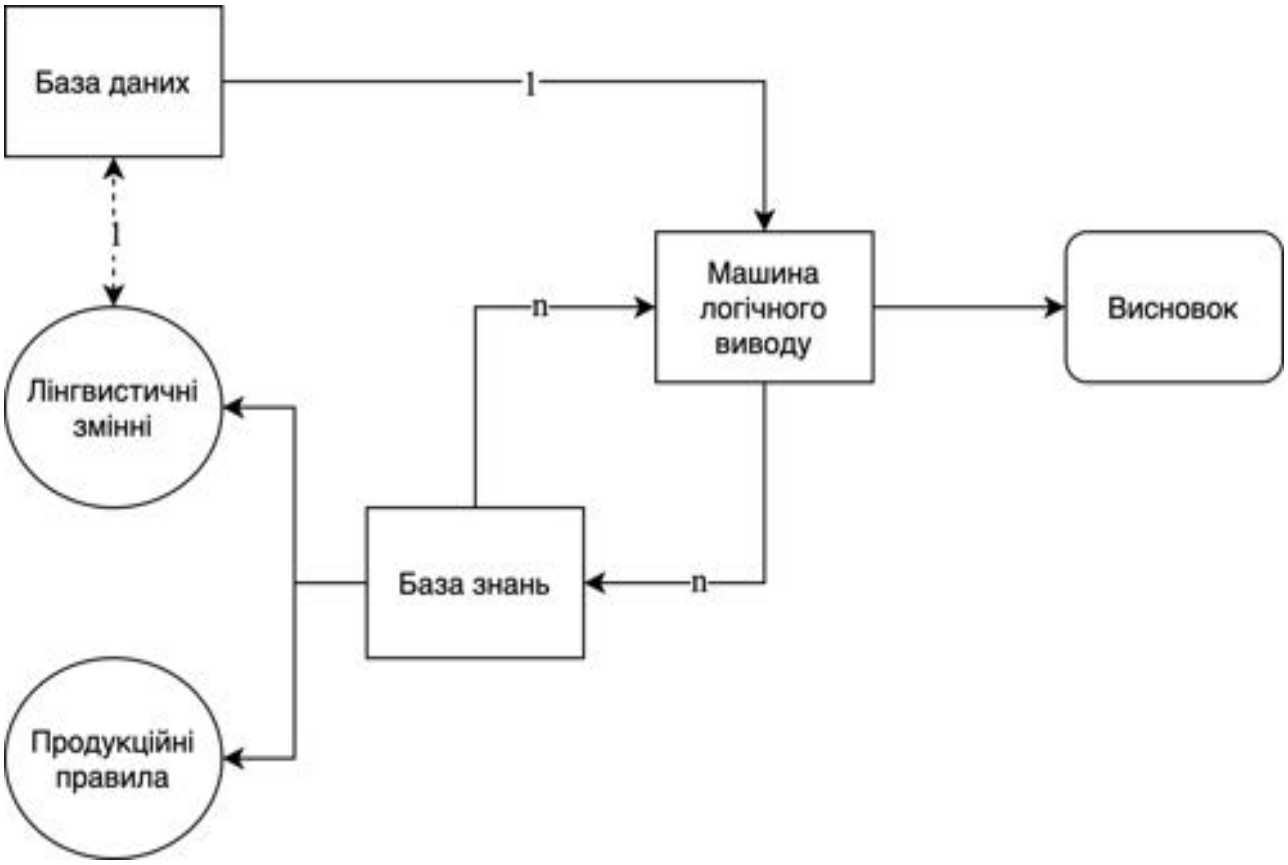


Рисунок 2.4 – Модифікований механізм продукційного виводу

(Джерело – власна розробка)

Розглянемо наступний приклад:

$$
\begin{aligned}
&R_1 : IF\ (ProjectType\ IS\ \ Telecommunication)\\
&THEN\ (ProjectStrategicGoal\ IS\ \ Aligned)
\end{aligned}
$$

$$
\begin{aligned}
&R_2 : IF\ (ProjectCompany\ IS\ \ NewCustomer\ \ \ AND\\
&\qquad ProjectCompanyRiskLevel\ IS\ Medium)\\
&THEN\ (ProjectCompanyStatus\ IS\ \ Cooperate)
\end{aligned}
\qquad (2.5)
$$

$$
\begin{aligned}
&R_3 : IF\ (ProjectStrategicGoal\ IS\ \ Aligned\ \ \ \ AND\\
&\qquad ProjectCompanyStatus\ IS\ \ Cooperate)\\
&THEN\ (ProjectTakeInProgress\ IS\ \ True)
\end{aligned}
$$

Правило R_1 є правило еквівалентності, що означає, що якщо попередник має значення TRUE, то послідовник також приймає значення TRUE. Правила

R_2 і R_3 є *i-infer* правилами, що означає, що кожен попередник повинен бути TRUE для того, щоб послідовники були TRUE.

Графи виводу були запропоновані і розроблені Шлегелем і Шапіро [92] як розширення пропозиційних графів. Граф виводу є графом міркувань, що здатний до зворотного, прямого і двонаправленого виводу. Він може підтримувати паралельну обробку для міркування з використанням продукційної логіки. Графи виводу модифікують пропозиційні графи, додаючи канали між вузлами уздовж можливих шляхів виводу. Канали несуть пріоритетні повідомлення для передачі нової інформації від одного вузла до іншого. Пріоритети повідомлень впливають на порядок виконання завдань для того, щоб спочатку виконувалися повідомлення, ближче до створення відповіді, і невідповідні завдання виводу були скасовані.

Вузол правила здатний виконувати операції виводу, використовуючи набір правил, відомих як "Інформація про використання правил" (RUI) [36]. Коли повідомлення надходить до вузла правила, повідомлення перетворюється на RUI. RUI містить інформацію про те, які попередники є істинними або помилковими (TRUE або FALSE), а також пояснює, як ці значення були виведені. Наново створена RUI поєднується з набором вже існуючих. Отримана комбінація використовується для визначення, чи можна застосовувати правила виводу вузла правила ще раз. Всі RUI, створені на вузлі, кешуються. Це запобігає повторному виведенню і скороченню циклів потоку повідомлень в графі, ігноруючи вже отримані RUI в кеші.

У розробленій системі використовується лише прямий вивід, тому структура графів виводу, запропонована Шлегелем і Шапіром, може бути значно спрощена:

- Потрібні лише два типи повідомлень: *i-infer* і *u-infer*. Повідомлення *i-infer* походять з нових вузлів, які були отримані як true або false, і надсилаються до вузлів правил, в яких поточний вузол є попередником. Повідомлення *u-infer* виникають у вузлах правил, які тільки отримали

всю необхідну інформацію про своїх попередників, вони повідомляють цільовому вузлу, який його новий статус.

- Немає необхідності в паралельних міркуваннях, тому зникає необхідність в пріоритетах для повідомлень.
- Замість RUI можемо використовувати простий статус правила, який оновлюється при надходженні нового повідомлення. Статус містить результат підрахунку статусів всіх попередників цього правила.

Враховуючи попередню інформацію продемонструємо семантичну мережу для набору правил (2.5).

Рис. 2.5 демонструє семантичну мережу з відповідними каналами. Ці канали дозволяють попередникам повідомляти вузли правила, коли вони були розраховані, а також дозволяють вузлам правил повідомляти про те, що вони були обчислені. Також продемонструємо процес виводу за цією семантичною мережею – рис. 2.6.

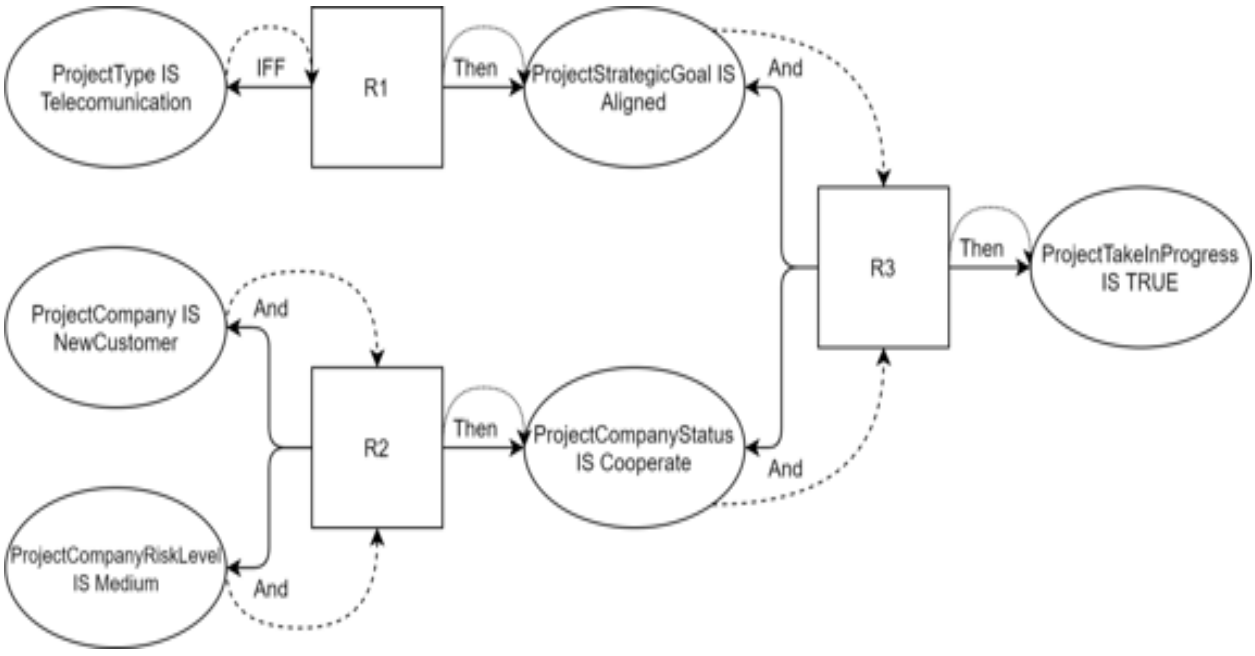


Рисунок 2.6 – Семантична мережа з каналами з набору правил (2.5)

(Джерело – власна розробка)

Фазифікація [68] є першим кроком у процесі роботи будь-якої нечіткої експертної системи, чіткі вхідні значення перетворюються в лінгвістичні значення зі ступенями належності до кожної з відповідних нечітких множин.

Фазифікація може бути досягнута за допомогою різних типів фазифікаторів, включаючи одношарові, гауссові, а також трапецієподібні або трикутні. Ці фазифікатори розраховують ступінь належності чіткого входу x до нечіткої множини A, використовуючи різні функції членства $\mu_A(x)$.

Фазифікувати − означає знайти ступінь приналежності лінгвістичних значень до змінної, що розглядається.

Як приклад розглянемо лінгвістичну змінну *ProjectCompanyRiskLevel*, значення якої можуть бути $x \in [0,35]$. Прикладами нечітких значень цієї змінної можуть бути значення *LOW, MEDIUM, HIGH* і вони є лінгвістичним еквівалентом деякого чіткого значення $x \in [0,35]$. Фазифікуємо цю змінну, використовуючи трапецієподібну функцією належності. Трапецієподібна функція визначається нижньою межею a, верхньою межею d, нижньою межею опори b, верхньою межею опори c, де $a < b < c < d$:

$$f(x,a,b,c,d) = \begin{cases} 0, & x \le a \\ \dfrac{x-a}{b-a}, & a \le x \le b \\ 1, & b \le x \le c \\ \dfrac{d-x}{d-c}, & c \le x \le d \\ 0, & d \le x \end{cases} \qquad (2.6)$$

У свою чергу, лінгвістичну змінну *ProjectCompanyRiskLevel* можна представити через наступні функції належності – рис. 2.7:

$$\mu(ProjectCompanyRiskLevel) = \begin{cases} LOW, & (0,5,10,15) \\ MEDIUM, & (10,15,20,25) \\ HIGH, & (20,25,30,35) \end{cases} \qquad (2.7)$$

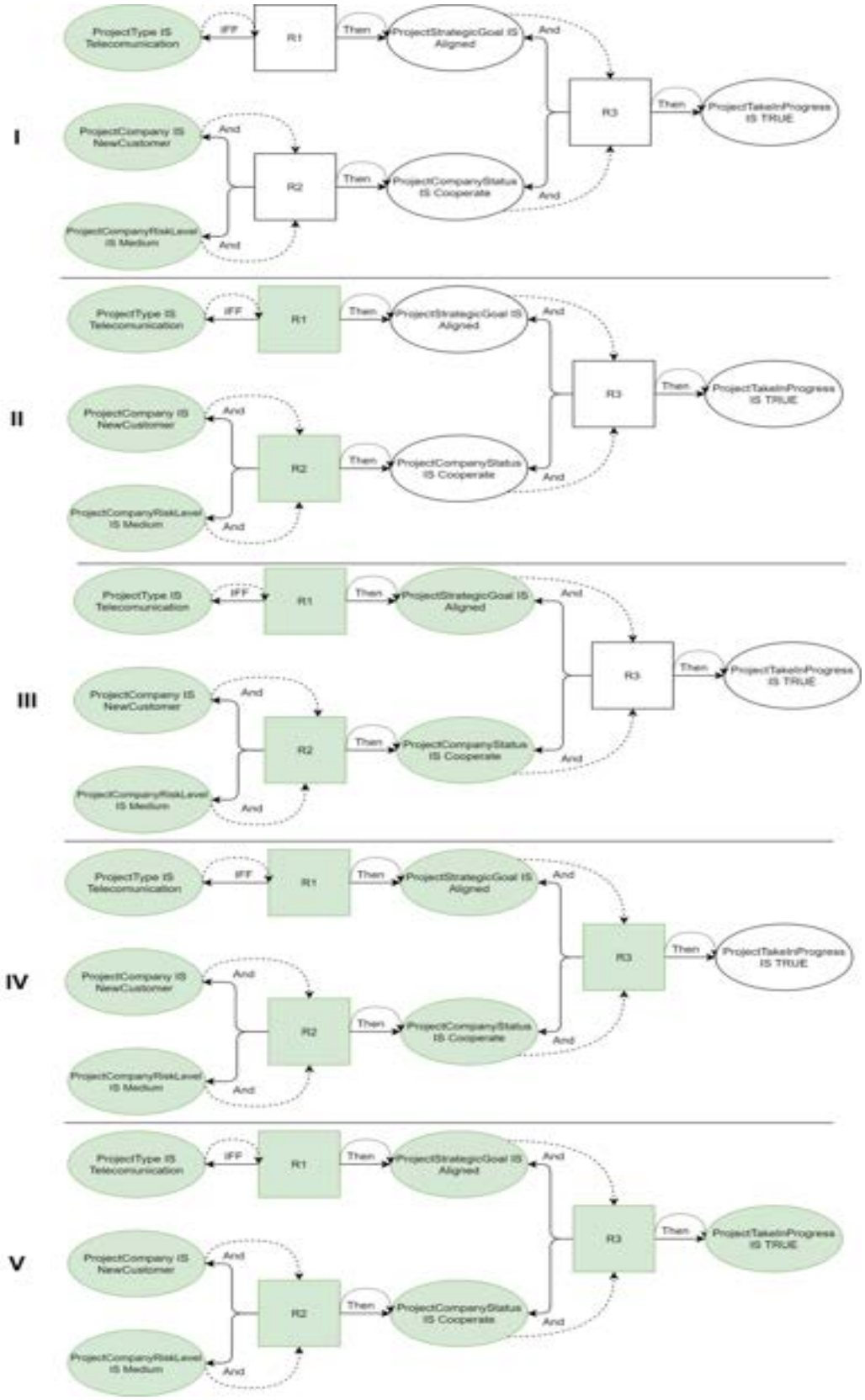


Рисунок 2.6 – Процес виводу з набору правил (2.5)

(Джерело – власна розробка)

З (2.6) та (2.7) випливають наступні значення функції належності лінгвістичної змінної *ProjectCompanyRiskLevel* для $x = 12$:

$$\mu_{LOW}(12) = \frac{d-x}{d-c} = \frac{15-12}{15-10} = 0.6$$
$$\mu_{MEDIUM}(12) = \frac{x-a}{b-a} = \frac{12-10}{15-10} = 0.4 \quad (2.8)$$
$$\mu_{HIGH}(12) = 0$$

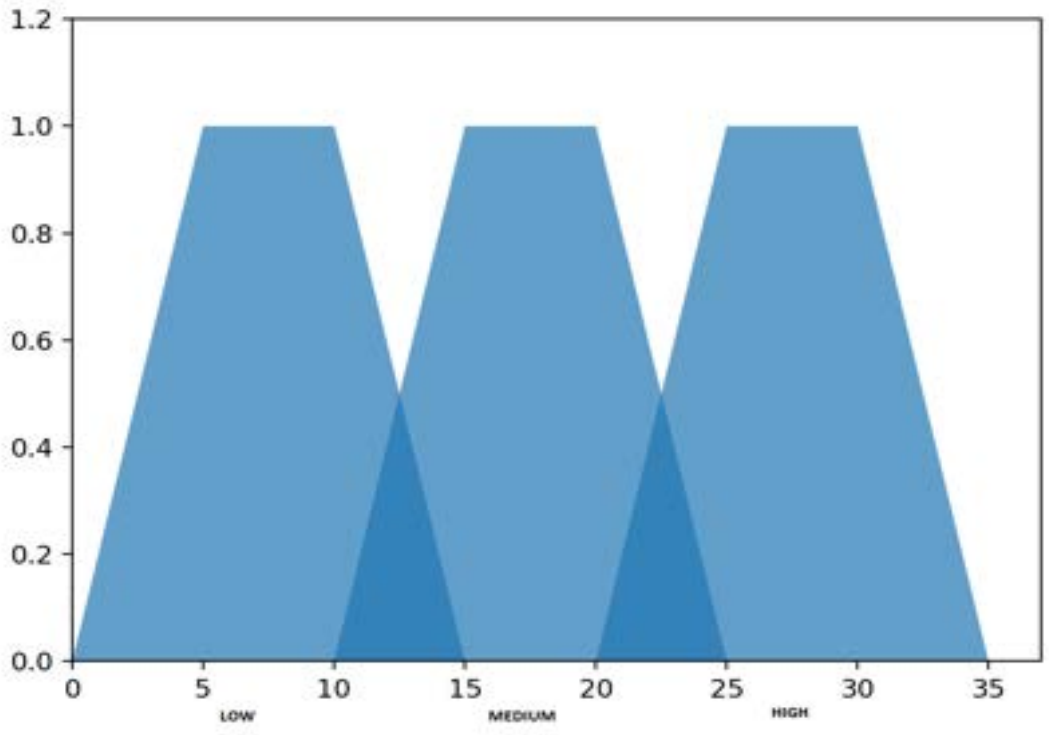


Рисунок 2.7 – Функції належності (2.7) у графічному представленні (Джерело – власна розробка)

Після фазифікації вхідних даних та отримання ступенів належності застосовуємо ці ступені до попередників нечітких правил. Якщо задане нечітке правило має декілька попередників, то нечіткий оператор (*AND* або *OR*) використовується для оцінки виразу. Нечіткий оператор *AND* використовується для оцінювання сполучення попередників правила. Наступна операція нечіткого перетину використовується для здійснення *AND* [124]:

$$\mu_{A\cap B}(u) = \min[\mu_A(u), \mu_B(u)] \quad (2.9)$$

Для прикладу розглянемо нечітке правило:

$$IF\ (ProjectCompany(X)\ \ IS\ \ NewCustomer\ \ AND$$
$$ProjectCompanyRiskLevel(Y)\ \ IS\ \ Medium)$$
$$THEN\ (ProjectCompanyStatus(Z)\ \ IS\ \ Cooperate)$$

Припустимо, що $\mu_{NewCustomer}(x) = 0.1$ та $\mu_{Medium}(y) = 0.6$. Тоді маємо:

$$\mu_{Cooperate}(z) = \min[\mu_{NewCustomer}(x), \mu_{Medium}(y)] = \min[0.1, 0.6] = 0.1$$

Подібним чином *OR* використовується для оцінки диз'юнкції попередників правила, яка реалізується класичним об'єднанням в нечітких логічних системах [124]:

$$\mu_{A \cup B}(u) = \max[\mu_A(u), \mu_B(u)] \qquad (2.10)$$

Для прикладу розглянемо попереднє нечітке правило з заміною *AND* на *OR*:

$$IF\ (ProjectCompany(X)\ \ IS\ \ NewCustomer\ \ OR$$
$$ProjectCompanyRiskLevel(Y)\ \ IS\ \ Medium)$$
$$THEN\ (ProjectCompanyStatus(Z)\ \ IS\ \ Cooperate)$$

Знову припустимо, що $\mu_{NewCustomer}(x) = 0.1$ та $\mu_{Medium}(y) = 0.6$. Тоді маємо:

$$\mu_{Cooperate}(z) = \max[\mu_{NewCustomer}(x), \mu_{Medium}(y)] = \max[0.1, 0.6] = 0.6$$

Тепер продемонструємо процес виводу по графу, враховуючи процес фазифікації вхідних даних. Припустимо, що після фазифікації вхідних даних маємо наступні значення: $\mu_{Telecomunication}(ProjectType) = 0.6$, $\mu_{NewCustomer}(ProjectCompany) = 0.8$, $\mu_{Medium}(ProjectCompanyRiskLevel) = 0.9$.

Таким чином послідовність виводу можна продемонструвати на рис. 2.8. Текстова репрезентація послідовності виводу графа:

1. Початкові значення:
   a. $\mu_{Telecomunication}(ProjectType) = 0.6$
   b. $\mu_{NewCustomer}(ProjectCompany) = 0.8$
   c. $\mu_{Medium}(ProjectCompanyRiskLevel) = 0.9$
2. Активація правил R_1 та R_2;

a. Похідні значення активованих правил R_1 та R_2:

b. $\mu_{Aligned}(ProjectStrategicGoal) = 0.6$

c. $\mu_{Cooperate}(ProjectCompanyStatus) = 0.8$

3. Активація правила R_3;

4. Висновок: $\mu_{True}(ProjectTakenInProgress) = 0.6$

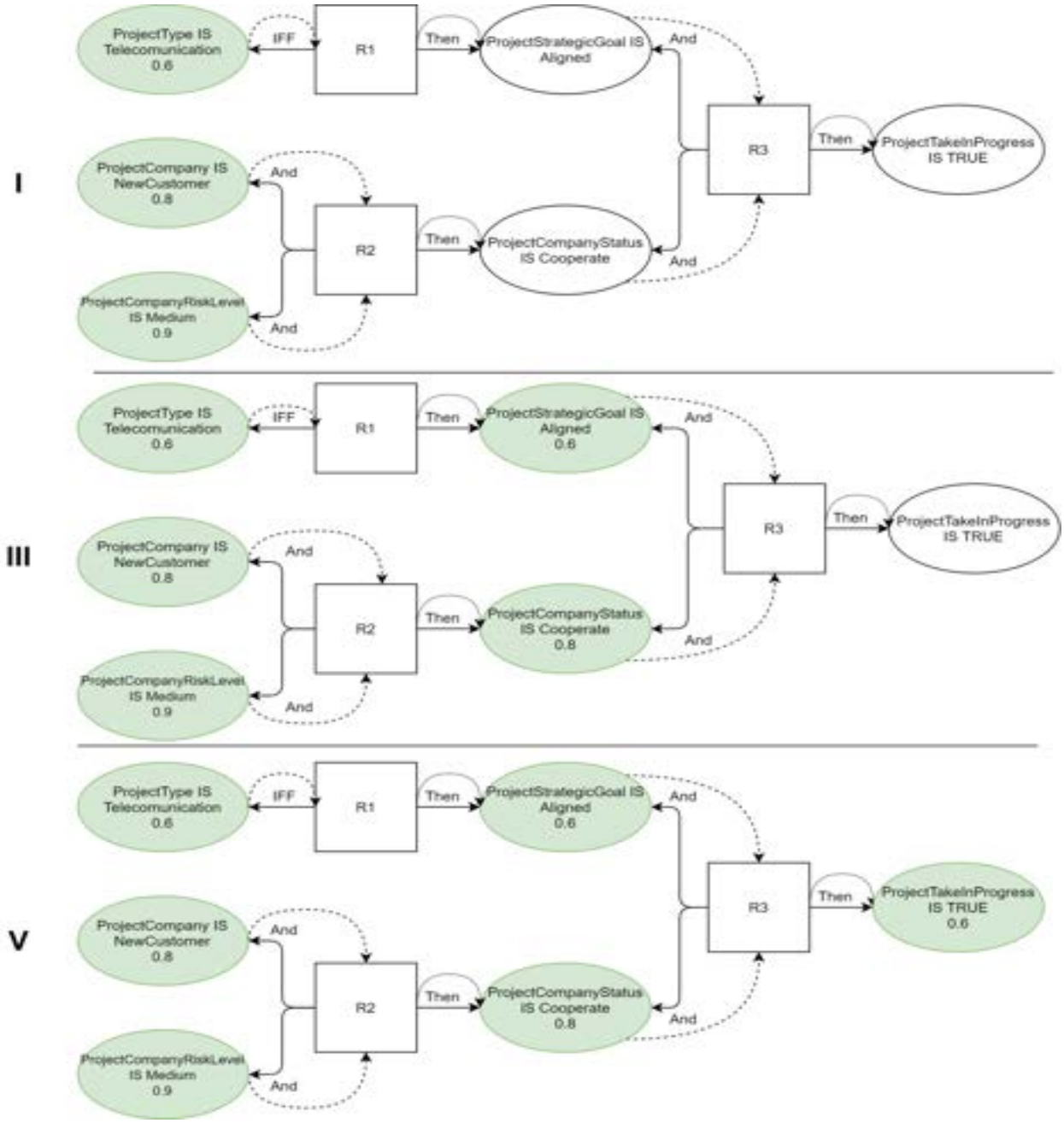


Рисунок 2.8 – Процес нечіткого виводу (Джерело – власна розробка)

Дефазифікація відбувається по закінченню процесу моделювання. Цей етап не є обов'язковим і використовується тільки тоді, коли спеціалісту необхідно перетворити нечіткий результат моделювання у чітке значення. Існує багато методів дефазифікації (COG, MOM і т.д.).

Програмну оболонку FuzzyKIDE було реалізовано з використанням COG (Center of Gravity) методу. Так як програмна оболонка має справу з дискретними функціями належності, то розрахунок чіткого значення виконується за формулою:

$$x' = \frac{\sum_{i=1}^{n} x_i' \cdot A_i}{\sum_{i=1}^{n} A_i},$$

де x_i' – локальний центр гравітації – як правило розраховується як середнє арифметичне точок відповідної лінгвістичної змінної у функції належності;

A_i – локальна область, визначена під лінією сегменту (p_{i-1}, p_i), що фактично є площею фігури, отриманої з точок відповідної лінгвістичної змінної у функції належності;

n – загальна (повна) кількість точок у відповідній лінгвістичній змінній у функції належності.

Знову розглянемо функцію належності лінгвістичної змінної *ProjectCompanyRiskLevel* (2.7). Використовуючи COG метод дефазифікації будемо мати наступні чіткі значення лінгвістичної змінної:

$$\mu_{LOW}(ProjectCompanyRiskLevel) = \frac{3.33 \cdot 2.5 + 7.5 \cdot 5 + 11.66 \cdot 2.5}{2.5 + 5 + 2.5} = 7.5$$

$$\mu_{MEDIUM}(ProjectCompanyRiskLevel) = \frac{13.33 \cdot 2.5 + 17.5 \cdot 5 + 21.66 \cdot 2.5}{2.5 + 5 + 2.5} = 17.5$$

$$\mu_{HIGH}(ProjectCompanyRiskLevel) = \frac{23.33 \cdot 2.5 + 27.5 \cdot 5 + 31.66 \cdot 2.5}{2.5 + 5 + 2.5} = 27.5$$

2.2.4 Архітектура оболонки FuzzyKIDE

Загальна архітектура розробленої оболонки FuzzyKIDE складається з наступних компонентів:

1. База даних (вхідні дані): зберігає початкові дані у вигляді значень, розділених комами (CSV). Кожне вхідне значення складається із своєї назви, через яку воно подається у базі знань, чіткого значення та коефіцієнта довіри від 0 до 1, який відображає впевненість користувача у чіткому значенні.
2. База знань: складається з продуційних правил та лінгвістичних змінних (розділ 2.2.2), які використовуються для міркування ядром логічного виводу.
3. Ядро виводу: займається обробкою вхідних даних на основі бази знань та реалізовано за допомогою семантичної мережі SNePS (розділ 2.2.3).
4. Графічний інтерфейс (GUI): дозволяє користувачеві взаємодіяти з базою знань, виконувати нечітке ядро виводу, переглядати нечіткі та чіткі результати виводу.

Мета графічного інтерфейсу – дозволити користувачам взаємодіяти з оболонкою FuzzyKIDE. Користувачі оболонки мають можливість працювати з базою знань, змінювати параметри виводу, завантажувати вхідні дані, виконувати нечіткий вивід та аналізувати результати виводу.

Графічний інтерфейс оболонки складається з 4 основних вікон. Початкове вікно – це форма входу \ реєстрації. Метою відокремити користувачів FuzzyKIDE для роботи з окремими базами знань є ціль спростити взаємодію користувача з оболонкою, завдяки чому користувачі з різними ролями не

перенавантажені інформацією. Після успішного входу \ реєстрації користувачі можуть отримати доступ до інших 3 основних вікон.

Одним з них є вікно для зміни параметрів виводу, наприклад, нижню межу коефіцієнта довіри. Цей параметр дозволяє ядру виводу встановити нижню межу довіри, при якій FuzzyKIDE рекомендує вживати заходів щодо запропонованих рішень.

Друге вікно призначене для роботи з багатьма базами знань шляхом додавання \ модифікації профілів, які складаються з продукційних правил та наборів лінгвістичних змінних. Графічний інтерфейс FuzzyKIDE дозволяє додавати профілі виводу, задаючи їм ім'я та опис. Після створення профілю FuzzyKIDE дозволяє додавати набори правил і набори змінних двома способами: додаючи їх поодинці напряму через інтерфейс або завантажуючи їх із файлів групами. Після додавання продукційних правил та лінгвістичних змінних користувачі мають можливість фільтрувати існуючі правила та змінні, а також змінювати їх у тому ж вікні.

Третє вікно призначене для проведення нечіткого виводу, як тільки користувач FuzzyKIDE вирішить, що база знань готова до використання для виводу. У цьому вікні користувач вибирає, який профіль (базу знань) використовувати, завантажує початкові дані (CSV) та запускає процес нечіткого виводу. Після закінчення виводу те саме вікно використовується для відображення та фільтрації результатів виводу. Пропозиції FuzzyKIDE можна також побачити на тому ж екрані, коли вибрано конкретний результат нечіткого виводу. Усі результати виводу оболонка представляє як в чіткому та і нечіткому вигляді.

Загальний процес взаємодії з FuzzyKIDE представлено на рис. 2.9. Далі розглянемо загальну технологію роботи з програмною оболонкою FuzzyKIDE на спрощеному прикладі прийняття рішення щодо взяття проєкту у розробку.

Ключову позицію в процесі роботи з розробленою типовою програмною оболонкою для роботи з нечіткою інформацією грає база знань.

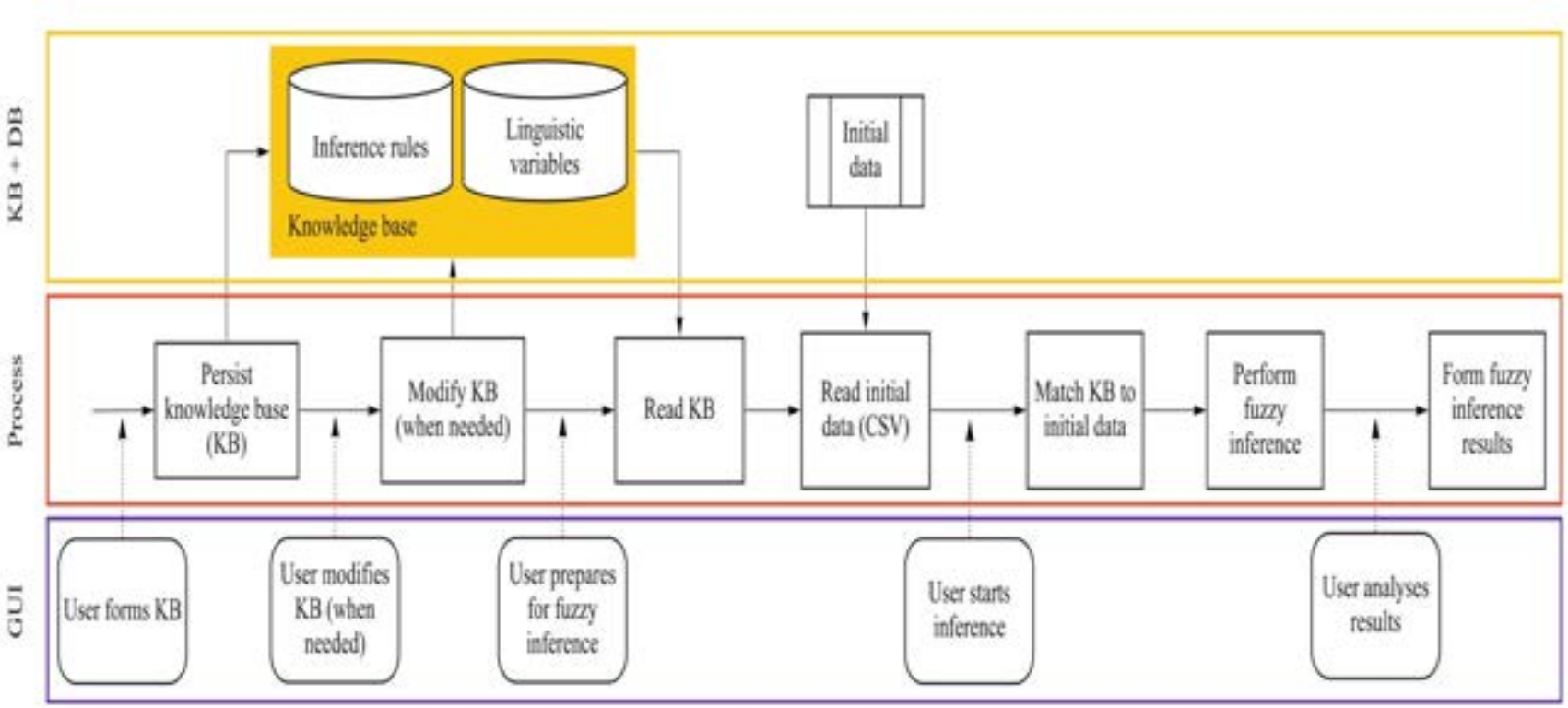


Рисунок 2.9 – Процес взаємодії з FuzzyKIDE

(Джерело – власна розробка)

База знань складається з двох основних частин:

- База нечітких правил;
- База лінгвістичних змінних, які використовуються у базі правил.

База нечітких правил складається з необмеженої кількості правил виду:

IF (ProjectCompany = NewCustomer & ProjectCompanyRiskLevel = Medium) THEN (ProjectCompanyStatus = Cooperate),

де *ProjectCompany, ProjectCompanyRiskLevel, ProjectCompanyStatu*s – лінгвістичні змінні, які задаються у базі лінгвістичних змінних;

NewCustomer, Medium, Cooperate – нечіткий лінгвістичний еквівалент деякого чіткого значення, пов'язаного з відповідною лінгвістичною змінною.

База лінгвістичних змінних складається з необмеженої кількості змінних виду:

[ProjectCompanyRiskLevel]:Initial:[Low:Trapezoidal:(0,5,10,15)|

Medium:Trapezoidal:(10,15,20,25)|High:Trapezoidal:(20,25,30,35)],

де *[ProjectCompanyRiskLevel]* – це набір лінгвістичних змінних (у даному випадку одна);

Initial/Derivative – позначає лінгвістичну змінну, значення якої буде взято, наприклад, з бази даних або значення якої буде виведено в процесі роботи з базою знань відповідно;

[Low:Trapezoidal:(0,5,10,15)|Medium:Trapezoidal:(10,15,20,25)|

High:Trapezoidal:(20,25,30,35)] – функції належності до нечіткого набору, а саме *Low, Medium, High* – набір значень для лінгвістичної змінної *ProjectCompanyRiskLevel, Trapezoidal* – трапецеїдальний тип функції належності, який використовується для опису значень лінгвістичної змінної та *(0,5,10,15), (10,15,20,25), (20,25,30,35)* – чіткі значення, які лежать за нечіткими значеннями Low, Medium, High відповідно.

Комбінація попереднього правила та лінгвістичних змінних, відповідних до цього правила виглядає наступним чином:

IF (ProjectCompany = NewCustomer AND ProjectCompanyRiskLevel = Medium) THEN (ProjectCompanyStatus = Cooperate),

[ProjectCompany]:Initial:[NewCustomer:Trapezoidal:(5,10,10,20)| ExistingCustomer:Trapezoidal:(15,20,20,25)],

[ProjectCompanyRiskLevel]:Initial:[Low:Trapezoidal:(0,5,10,15)| Medium:Trapezoidal:(10,15,20,25)| High:Trapezoidal:(20,25,30,35)],

[ProjectCompanyStatus]:Derivative:[Cooperate:Trapezoidal:(100,200,200,600), Medium:Trapezoidal:(10,15,20,25)| Reject:Trapezoidal:(550,1000,1100,1500)].

Кількість правил та лінгвістичних змінних у типовій оболонці є необмеженою, але треба мати на увазі, що їх комбінація завжди має бути валідною:

1. Нечіткі правила повинні відповідати попередньо розглянутому вигляду. Програмна оболонка розроблена таким чином, що нечіткі правила не відповідаючі цьому вигляду неможливо додати до бази знань за причиною невалідності;
2. Лінгвістичні змінні повинні відповідати попередньо розглянутому вигляду. Програмна оболонка розроблена таким чином, що лінгвістичні змінні не відповідаючі цьому вигляду неможливо додати до бази знань за причиною невалідності;
3. Завжди має існувати чітка відповідність нечіткого правила до лінгвістичних змінних, а саме, якщо правило задає деяку лінгвістичну змінну, вона має бути описана в базі лінгвістичних змінних. Програмна оболонка розроблена таким чином, що у іншому випадку база знань вважається невалідною, що унеможливлює подальшу роботу з нею.

Початкові данні необхідні для старту процесу нечіткого виводу у ядрі виводу та мають наступний вигляд: *ProjectCompany:9:0.7*,

де *ProjectCompany* – ім'я лінгвістичної змінної, значення якої задають ці початкові данні;

9 – чітке початкове значення лінгвістичної змінної;

0.7 – коефіцієнт достовірності, який приймає значення [0,1] та показує на скільки користувач оболонки впевнений у початкових даних.

Для попередньо розглянутої комбінації нечіткого правила та відповідних лінгвістичних змінних початкові данні мають наступний вигляд:

ProjectCompany:9:0.7,

ProjectCompanyRiskLevel:13:0.6.

Для лінгвістичної змінної *ProjectCompany* чітке значення 9 відповідає нечіткому значенню *NewCustomer*, а у лінгвістичної змінної *ProjectCompanyRiskLevel* чітке значення 13 відповідає нечіткому значенню *Medium*.

У поточній реалізації програмної оболонки було обрано найпростіший спосіб збереження та роботи з початковими даними – використовуючи текстові CSV файли. Кількість початкових даних є необмеженою, але треба мати на увазі, що комбінація початкових даних та бази знань має бути валідною для початку роботи процесу нечіткого виводу у ядрі виводу, а саме кожній лінгвістичній змінній з поміткою Initial має відповідати запис з файлу початкових даних. У іншому випадку неможливо розпочати процес нечіткого виводу через недостачу початкових даних.

Якщо комбінація бази знань та початкових даних є валідною, то користувач оболонки може почати процес нечіткого виводу. Також, необхідно мати на увазі, що особливості роботи ядра нечіткого виводу оболонки залишають можливість для роботи з базами знань різноманітних прикладних областей, що робить оболонку типовою та достатньо гнучкою для найрізноманітніших завдань.

2.3 Методичні засади та технологія роботи з нечіткою експертною системою FuzzyKIDE

Першим етапом є формування бази знань та початкових даних. Далі наведені дані є продовженням прикладу, продемонстрованого в розділі 2.2.

Почнемо з нечітких правил:

IF (ProjectType = Telecommunication) THEN (ProjectStrategicGoal = Aligned),

IF (ProjectCompany = NewCustomer & ProjectCompanyRiskLevel = Medium) THEN (ProjectCompanyStatus = Cooperate),

IF (ProjectStrategicGoal = Aligned & ProjectCompanyStatus = Cooperate) THEN (ProjectTakeInProgress = True),

Продовжимо опис лінгвістичними змінними:

[ProjectType]:Initial:[Telecommunication:Trapezoidal:(0,20,20,30)| SinglePage:Trapezoidal:(20,40,60,80)| TravelAndEvents:Trapezoidal:(70,80,80,90)],

[ProjectStrategicGoal]:Initial:[Misaligned:Trapezoidal:(0,50,50,100)| Aligned:Trapezoidal:(50,150,150,200)],

[ProjectCompany]:Initial:[NewCustomer:Trapezoidal:(5,10,10,20)| ExistingCustomer:Trapezoidal:(15,20,20,25)],

[ProjectCompanyRiskLevel]:Initial:[Low:Trapezoidal:(0,5,10,15)| Medium:Trapezoidal:(10,15,20,25), High:Trapezoidal:(20,25,30,35)],

[ProjectCompanyStatus]:Initial:[Cooperate:Trapezoidal:(100,200,200,600)| Medium:Trapezoidal:(10,15,20,25)| Reject:Trapezoidal:(550,1000,1100,1500)],

[ProjectTakeInProgress]:Derivative:[False:Trapezoidal:(0,25,25,49)| True:Trapezoidal:(50,100,100,150)].

Завершимо опис початковими даними з CSV файлу:

ProjectType:25:0.8,
ProjectCompany:9:0.7,
ProjectCompanyRiskLevel:13:0.6.

З початкових даних маємо значення типу проєкту *ProjectType = 25*, що відповідає проєктам типу Telecommunication або *SinglePage*, з досить високим коефіцієнтом достовірності 0.8. Також маємо значення типу компанії

ProjectCompany = 9, що відповідає нечіткому значенню *NewCustomer*, з високим коефіцієнтом достовірності 0.7.

Наприкінці маємо рівень ризику *ProjectCompanyRiskLevel = 13*, що відповідає *Low* та *Medium* рівням ризику та середнім коефіцієнтом достовірності 0.6, що свідчити про невпевненість користувача щодо цього значення.

Тепер перейдемо до покрокового розгляду, яким чином користувач може почати роботу з попередньо описаними нечіткими правилами, лінгвістичними змінними та початковими даними через інтерфейс для роботи з програмною оболонкою.

Спочатку користувачеві необхідно зареєструватися або виконати вхід у програмний додаток, натиснувши на вкладку **«Log In»**. Це робиться з метою розділити список задач, що підлягають розв’язанню, між різними користувачами програмного додатку, щоб не змішувати їх бази знань.

Процес додавання та процес виконання входу користувача продемонстровано на рис. 2.10.

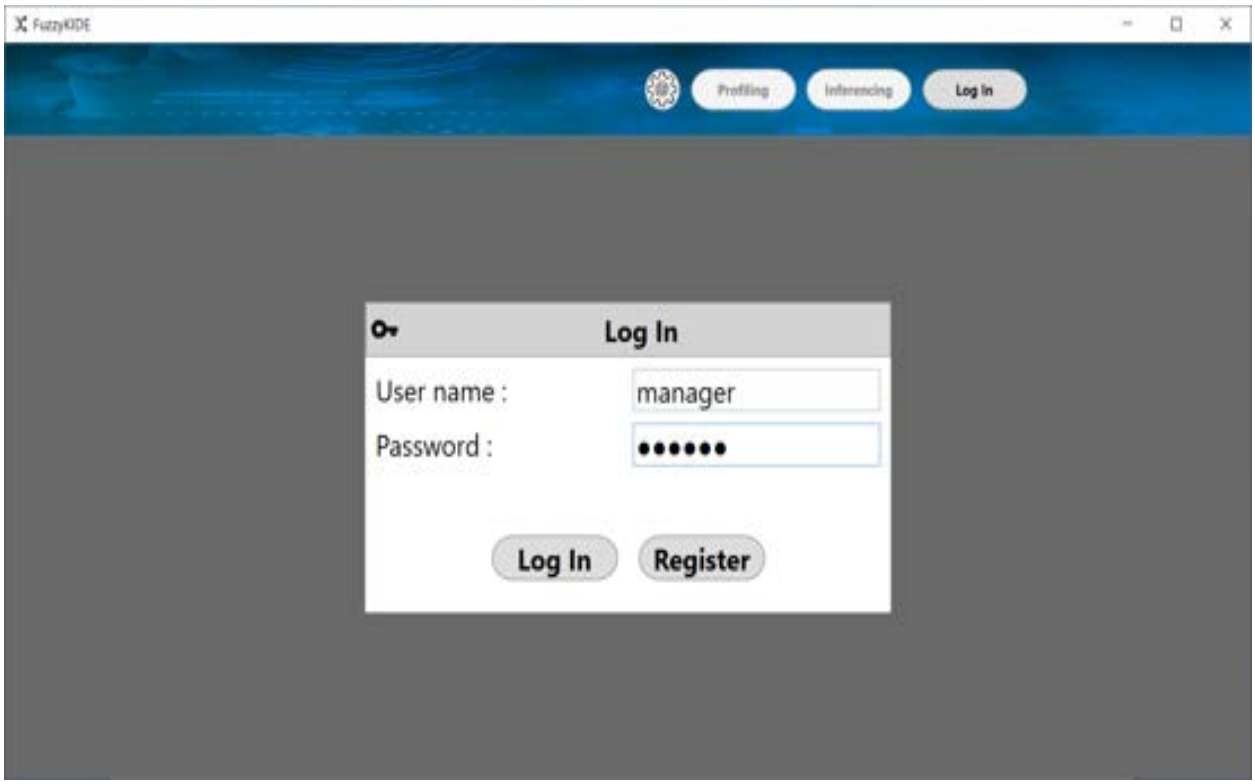


Рисунок 2.10 – Додавання користувача (Джерело – власна розробка)

Далі розглянемо процес роботи з базою знань для користувача, який виконав вхід в програмний додаток. Весь процес роботи з базою знань зосереджено у вкладці **«Profiling»**. Для спрощення роботи з базою знань, програмна оболонка дозволяє працювати з нею у форматі профілів. Кожен з профілів відповідає окремій задачі, з якою може працювати програмна оболонка.

Процес додавання профілю продемонстровано на рис. 2.11 на прикладі додання профілю *Project Selection*.

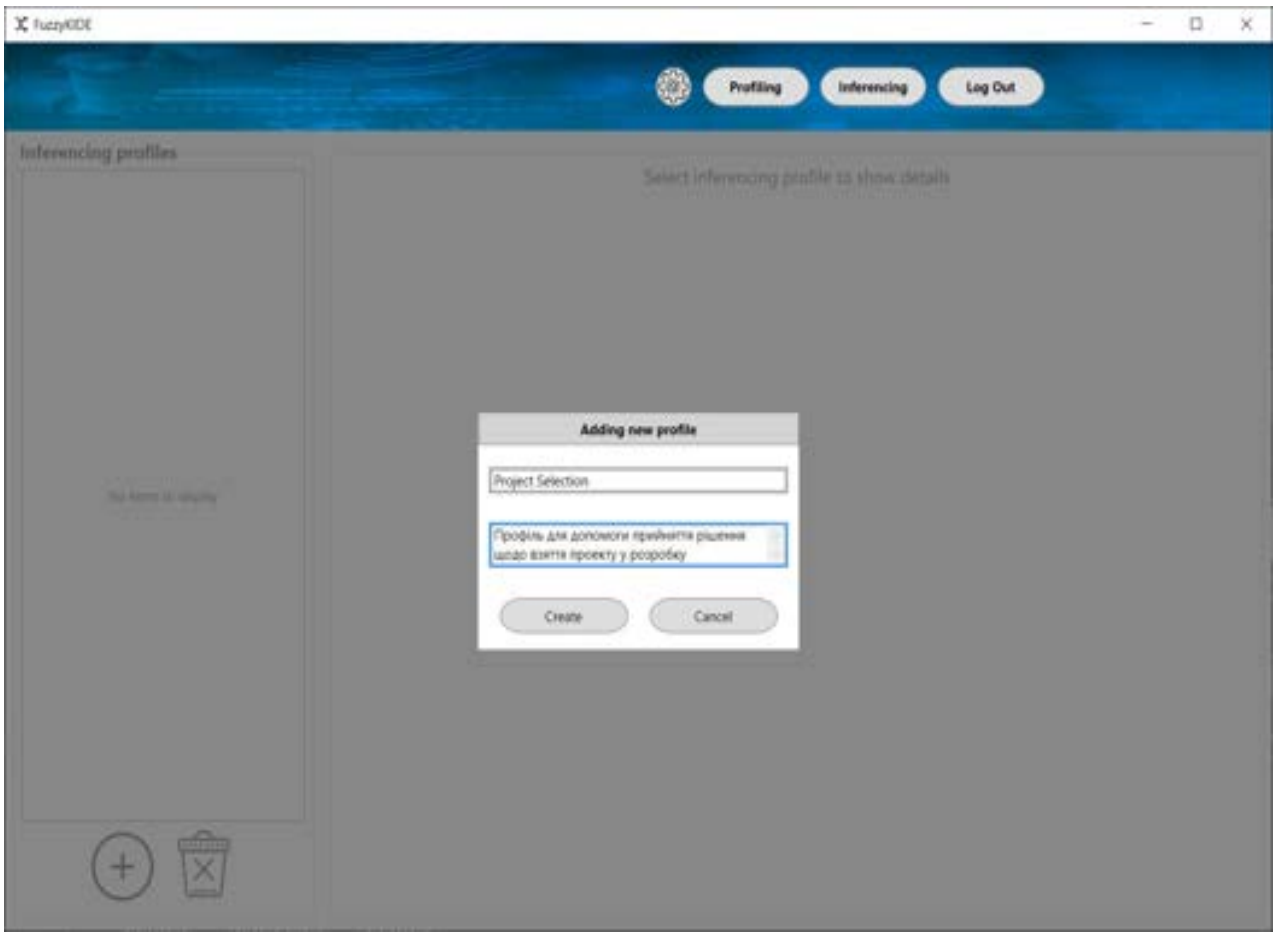


Рисунок 2.11 – Додавання профілю (Джерело – власна розробка)

При додаванні профілю йому необхідно дати ім'я та короткий опис. У результаті у списку профілів з'явиться профіль *Project Selection* – рис. 2.12.



Рисунок 2.12 – Список профілів (Джерело – власна розробка)

Натиснувши на профіль Project Selection перейдемо до роботи з базою знань цього профілю. У цьому режимі у користувача є можливість переглянути базову інформацію профілю, ініціалізувати / оновити базу знань профіль, або видалити профіль натиснувши на кнопку у формі корзини.

Для додавання нечітких правил та лінгвістичних змінних необхідно натиснути **«Update»**. Після цього з’являється вікно, яке дає користувачеві дві опції – рис. 2.13:

- Додавати правила та змінні поодинці через поле вводу;
- Додавати правила та змінні у необмежених кількостях з окремих файлів, у яких збережені нечіткі правила та лінгвістичні змінні.

Для набору нечітких правил та відповідних лінгвістичних змінних з цього розділу маємо ситуацію, продемонстровану на рис. 2.14. Після додання нечітких правил та лінгвістичних змінних у користувача є можливість фільтрувати та модифікувати уже наявну базу знань – рис. 2.15.

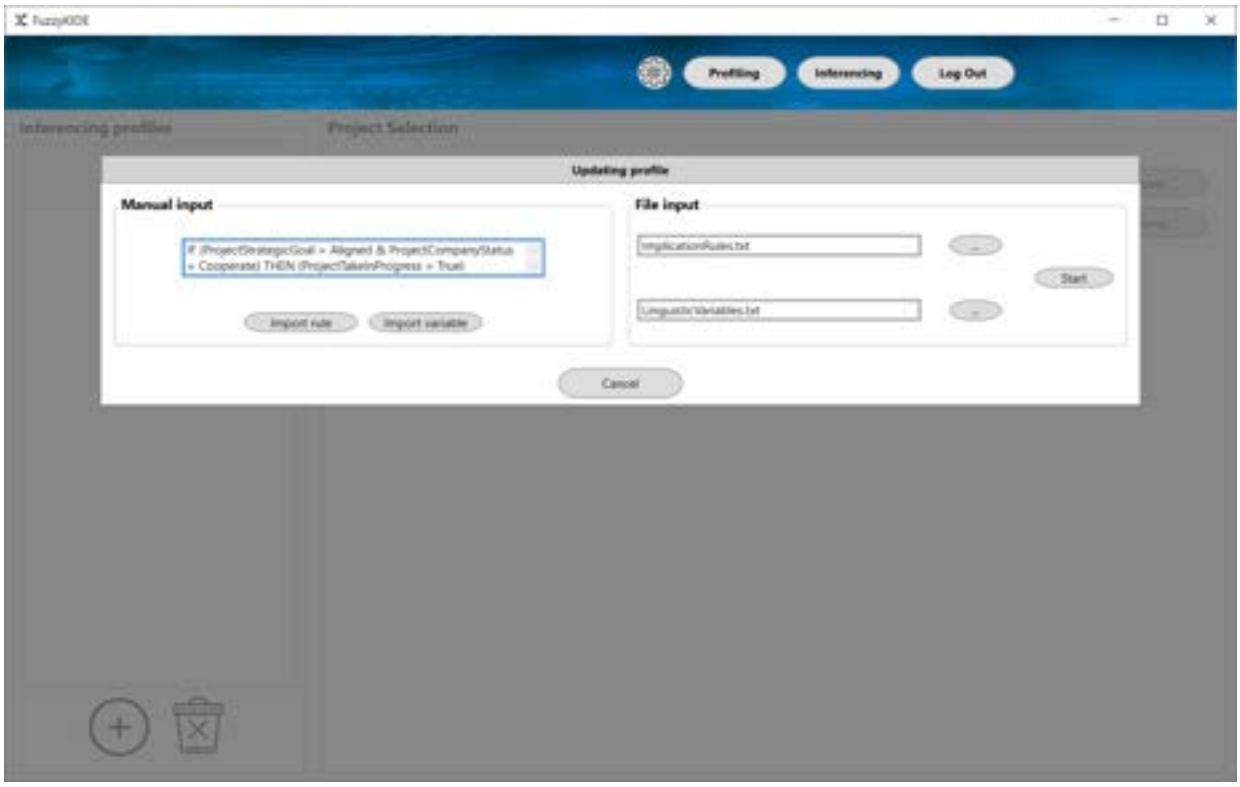


Рисунок 2.13 – Додавання інформації до бази знань профілю

(Джерело – власна розробка)

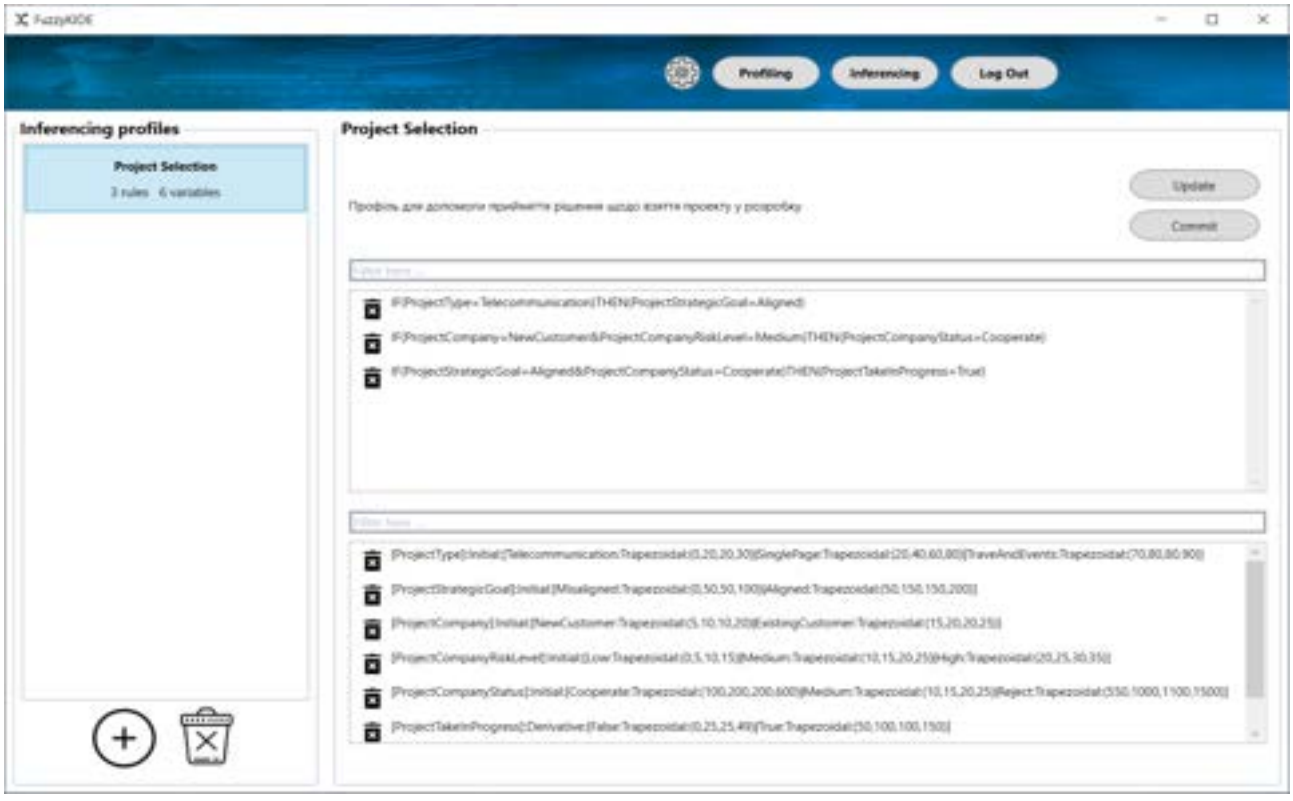


Рисунок 2.14 – Огляд бази знань профілю

(Джерело – власна розробка)

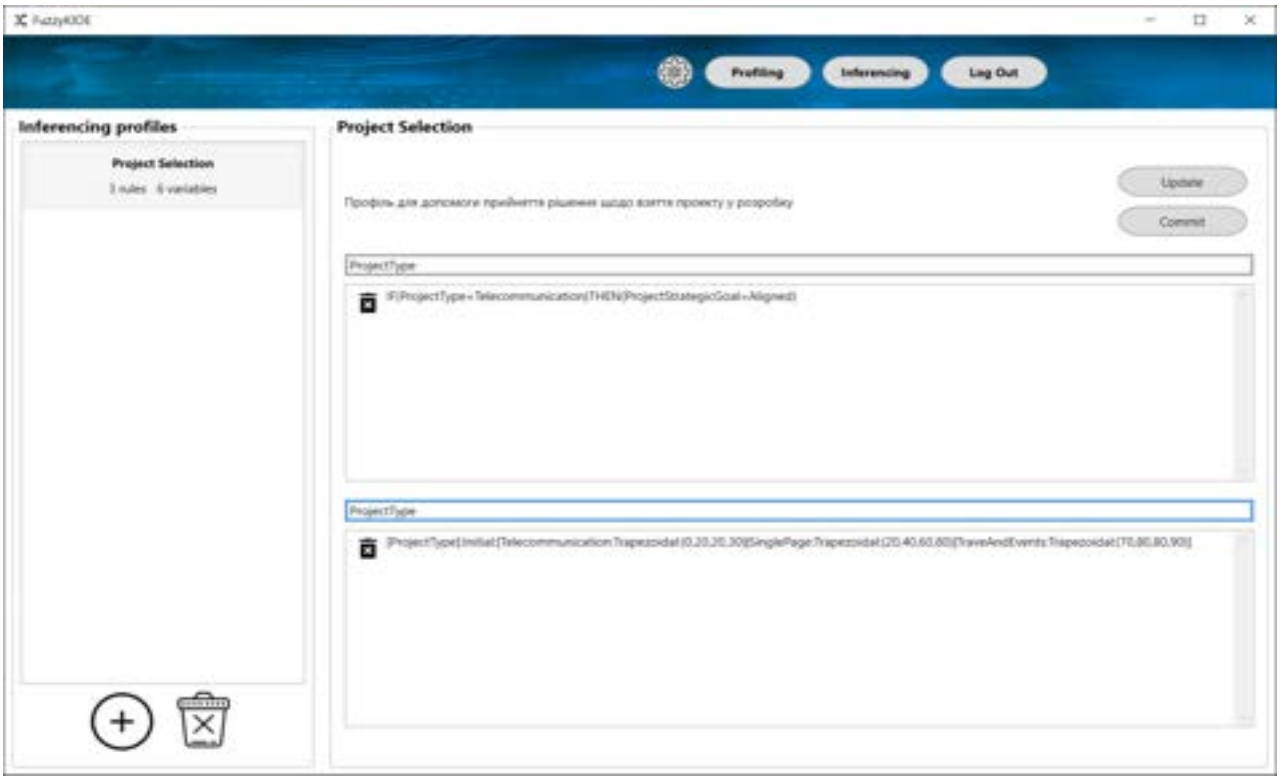


Рисунок 2.15 – Фільтрація бази знань профілю

(Джерело –власна розробка)

Натиснувши на профіль Project Selection перейдемо до роботи з базою знань цього профілю. У цьому режимі у користувача є можливість переглянути базову інформацію профілю, ініціалізувати / оновити базу знань профіль, або видалити профіль натиснувши на кнопку у формі корзини.

Для додавання нечітких правил та лінгвістичних змінних необхідно натиснути **«Update»**. Після цього з’являється вікно, яке дає користувачеві дві опції – рис. 2.13:

- Додавати правила та змінні поодинці через поле вводу;
- Додавати правила та змінні у необмежених кількостях з окремих файлів, у яких збережені нечіткі правила та лінгвістичні змінні.

Для набору нечітких правил та відповідних лінгвістичних змінних з цього розділу маємо ситуацію, продемонстровану на рис. 2.14. Після додання нечітких правил та лінгвістичних змінних у користувача є можливість фільтрувати та модифікувати уже наявну базу знань – рис. 2.15.

Після завершення роботи з базою знань, користувачеві необхідно натиснути **«Commit»** для її збереження у внутрішню базу знань програмної оболонки для подальшого її використання після припинення роботи оболонки та для використання у процесі нечіткого виводу.

Далі розглянемо процес роботи з ядром виводу. Цей процес зосереджено у вкладці **«Inferencing»**.

Для початку роботи з ядром виводу спершу необхідно обрати базу знань, завчасно сформовану під час роботи з базою знань. Після цього необхідно обрати CSV файл з початковими даними.

Після обрання профілю бази знань у одному з вікон буде виведено скорочену інформацію щодо цього профілю. Після завдання профілю бази знань та початкових даних стає доступна опція **«Start Inference»**, що стартує нечіткий вивід.

По завершенню виводу у правій частині вікна вкладки виводу можна переглянути хід активації унарних вузлів нечітких правил з ціллю виявити необхідний результат виводу.

Також після цього доступна опція **«View All Results»**, яка відкриває текстовий редактор, який демонструє усі використані нечіткі правила та повний процес нечіткого виводу, що дозволяє виконувати більш зручний пошук в разі необхідності – рис. 2.16.

Проаналізуємо результат нечіткого виводу на основі профілю бази знань *Project Selection*. Користувача програмної оболонки цікавлять відомості щодо активування вузла *ProjectTakeInProgress = True (100)*, який показує чи є сенс брати проєкт в роботу чи ні. Як бачимо з результатів виводу цей вузол було активовано з коеффіцієнтом достовірності 0.6. Активізація вузла свідчить про достатню кількість даних для прийняття рішення щодо взяття проєкту у розробку. Але так як порогове значення складає 0.6, що менше порогового значення за замовчуванням (0.8).

Це свідчить про невпевненість експертної системи у початковому наборі даних, які використовувалися для активації цього вузла.

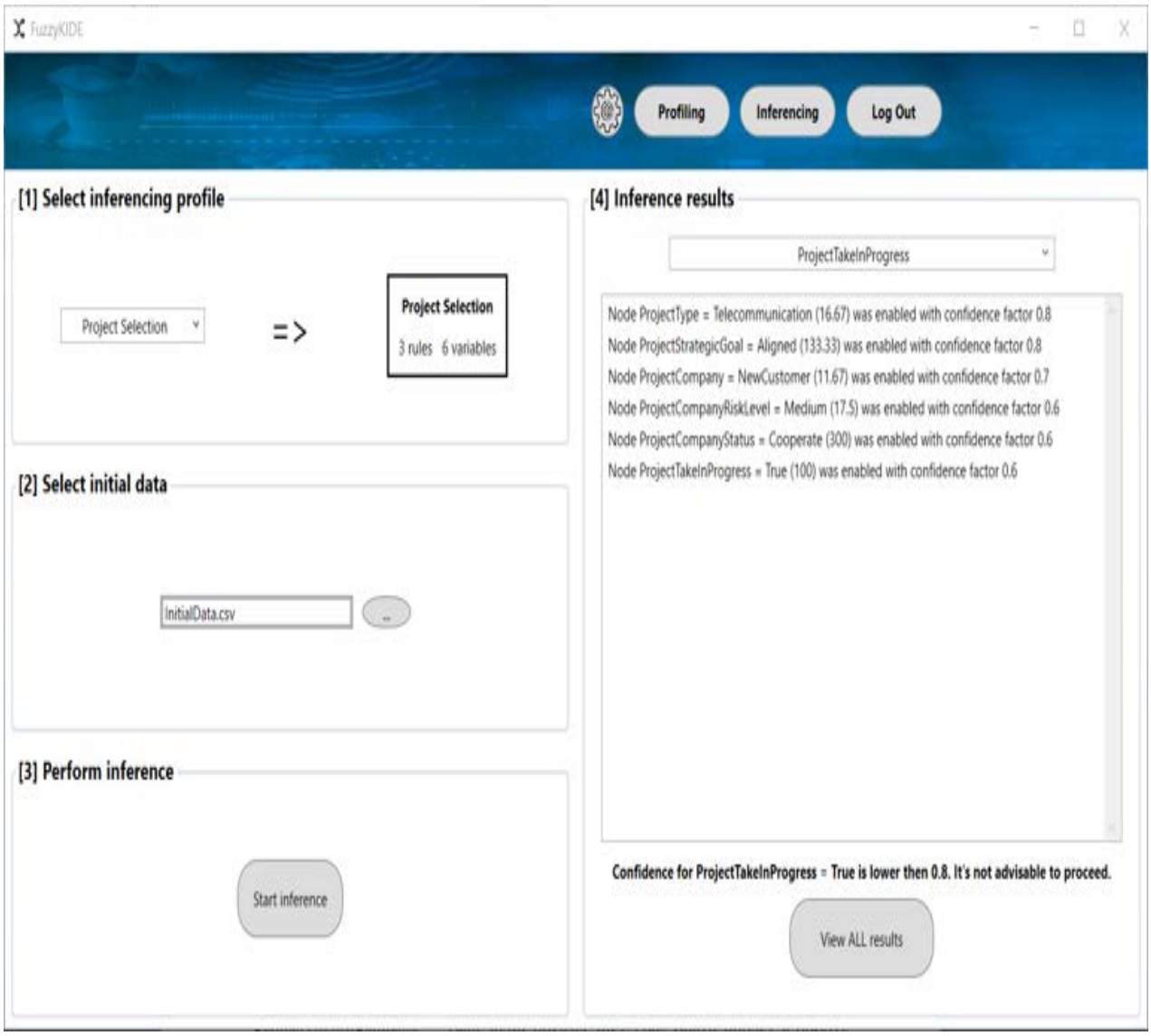


Рисунок 2.16 – Результат роботи з ядром виводу

(Джерело – власна розробка)

Іншими словами, експертна система показує, що є сенс брати проєкт у розробку, але перед цим експерту бажано додатково проаналізувати ризики.

Також у круглих дужках можемо бачити розраховане чітке значення (100), що відповідає нечіткому значенню *True* лінгвістичної змінної *ProjectTakeInProgress*.

У разі неприйнятності порогового значення за замовчуванням (0.8) при нечіткому виводі, програмна оболонка дає можливість замінити його на інше значення натиснувши на круглу вкладку з малюнком шестерні – рис. 2.17.

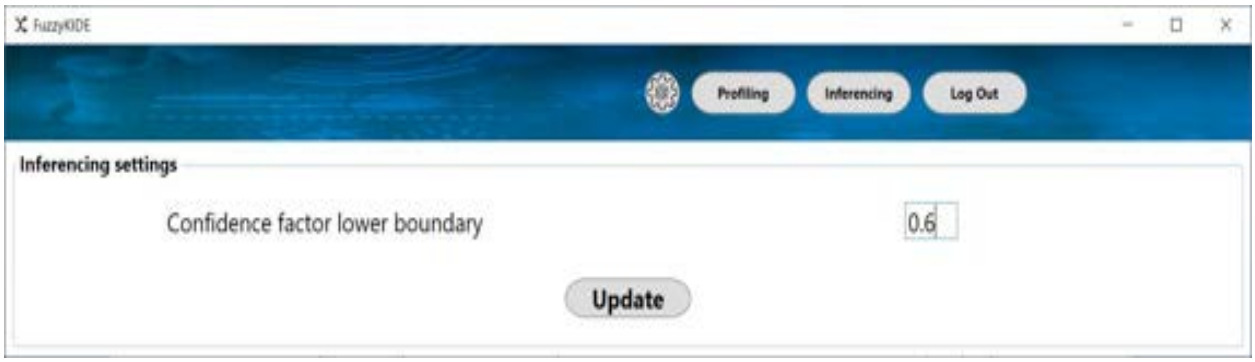


Рисунок 2.17 – Налаштування параметрів програмної оболонки
(Джерело – власна розробка)

Якщо у користувача є необхідність передати програмну оболонку іншому експерту, то після завершення роботи зі своїми профілями користувач може настиснути **«Log Out»**, що дозволяє передати оболонку, не закриваючи її.
Як можемо бачити, інтерфейс оболонки розроблено без прив'язування до прикладних задач специфічних предметних областей, що в підтримку до типовості процесу створення бази знань тільки підсилює її можливість використання для вирішення різноманітних завдань з різноманітних прикладних областей.

2.4 Модель Agile Stage-Gate фреймворку з використанням нечіткої логіки: зміст, функціонал та переваги

Розглянемо шляхи усунення труднощів, пов'язаних з використанням Stage-Gate фреймворку та Agile технології, продемонструвавши, їх комбінацію, яка враховує особливості IT-проєктів в цілому та особливості сучасних ітеративних циклів розробки.

Основна причина відмінностей полягає у різних цілях двох технологій. Stage-Gate – це всеосяжна технологія запуску нових проєктів через

макропланування, а Agile – технологія управління процесом розробки проєктів через мікропланування [37]. Розглянемо порівняння двох технологій у табл. 2.1.

Таблиця 2.1 – Порівняння Agile та Stage-Gate фреймворк технологій (Джерело – власна розробка)

	Stage-Gate	**Agile**
1	2	3
Тип	макропланування	мікропланування, управління проєктами
Обсяг	від ідеї до запуску	розробка та тестування, може бути розширена до фази попередньої розробки
Організація	багатофункціональна команда (R&D, маркетинг, продаж, операції)	технічна команда (розробники програмного забезпечення, інженери)
Модель прийняття рішення	інвестиційна модель – у критичних рішеннях задіяна вища група управління	тактична модель – рішення щодо дій щодо наступного спринту, що в значній мірі приймаються самокерованою командою

Stage-Gate є багатофункціональним, тобто залучає людей, які займаються маркетингом та технічний персонал, і має кілька етапів, що охоплюють весь процес від відкриття ідеї нового продукту до запуску цього продукту. Рішення в Stage-Gate технології відповідають моделі інвестиційних рішень – рішення про перехід через брами впливає на перерозподіл ресурсів на проєкти в міру проявлення їх потенціалу. Таким чином, Stage-Gate дає вказівки щодо того, які проєкти потрібно розробляти, а потім що робити в рамках кожного проєкту.

З іншого боку, Agile технологія розроблена спеціально для того, щоб допомогти розробникам продуктів швидко створювати робоче програмне забезпечення з постійною валідацією від замовника. Після того, як проєкт розробки був затверджений і його початкові вимоги були накреслені, Agile

акцентує увагу на процесі програмної розробки. Кожна ітерація процесу програмної розробки може не виробляти достатньо функціональних можливостей, щоб гарантувати випуск на ринок, але початкова мета – мати продукт, який потенційно готовий до випуску у кінці кожної ітерації.

Таким чином, перша відмінність полягає у тому, що технологія використовується в основному під час етапів розробки та тестування проєкту, тобто охоплюють два етапів із п’яти чи шести, що входять до типового процесу Stage-Gate фреймворку. І його в основному використовують технічні спеціалісти, які фактично займаються розробкою програмного забезпечення. Друга концептуальна відмінність між двома технологіями полягає у різній точці зору щодо основних змінних в управлінні проєктами: обсягу роботи, бюджету та тривалості виконання. У Stage-Gate фреймворці обсяг робіт фіксований, а бюджет і час – гнучкі. Але в Agile методології, за кожним спринтом фіксуються тривалість виконання і бюджет, а обсяг роботи може змінюватися – табл. 2.2.

Таблиця 2.2 – Порівняння Agile та Stage-Gate фреймворк технологій (Джерело –власна розробка)

Показники	**Stage-Gate**	**Agile**
1	2	3
Характеристика	орієнтований на планування	орієнтований на споживача, планування на льоту
Обсяг	фіксований	гнучкий
Бюджет	гнучкий	фіксований
Час	гнучкий	фіксований

В Agile практиці є кілька великих IT-фірм, які почали вбудовувати у свої існуючі Stage-Gate фреймворк процеси створення програмного забезпечення Agile технологію, таким чином створюючи гібридний фреймворк створення програмного забезпечення. Їх досвід свідчить про те, що Agile та Stage-Gate

можна використовувати разом для отримання більшої користі ніж окремо. У дослідженні [37, 116] розкриваються подробиці переходу від класичного Stage-Gate фреймворку в компаніях Ericsson та Vodafone до Agile Stage-Gate фреймворку. Це поєднання дало декілька основних переваг:

1. *Покращене розуміння проєкту:* ітеративний процес розробки програмного забезпечення, дозволяє замовникові швидко та більш дешево отримати проміжковий результат; менеджменту на багато простіше слідкувати за змінами в процесі;
2. *Більше можливостей для контролю невизначеності:* повне розуміння вимог до продукту є частиною ітеративних циклів розробки, тому значно більша вірогідність, що перед початком етапу розробки, вимогу будуть сформульовані конкретні вимоги, так як скоріше за все вже будуть готові попередні тестові версії;
3. *Прискорений процес розробки:* команда чітко знає ступінь зобов'язань протягом кожного спринту з заздалегідь обговореними термінами на виконання цих зобов'язань;
4. *Більше фокусу на проєкт:* як правило, команди працюють над розробкою тільки одного проєкту, що призводить до збільшення концентрації команди над проєктом та покращення якості проєкту;

Більше комунікації: також покращення якості проєкту відбувається за рахунок покращення шляхів комунікації в команді та між командою і менеджментом, що є невід’ємною частиною Agile технологій розробки.

Результат поєднання Agile та Stage-Gate продемонстровано на рис. 2.18.

Найбільш важливі результати випливають із двох факторів.

Правильне розуміння продукту. Численні дослідження виявляють, що багато нових продуктів не дають обіцяної цінності при впровадженні на ринок. Основна причина невдачі: нерозуміння потреб клієнтів та невідповідності товару для ринку. Навіть незважаючи на те, що компанії, які використовують Stage-Gate фреймворк, мають більш високий рівень успішності, все ще є можливість для вдосконалення. Використання ітеративних спринтів, які

інтегрують в собі численні тести замовника та валідацію на ринку значно збільшує шанси на отримання продукту, в якому є потреба і тим самим значно збільшує шанси на успіх.

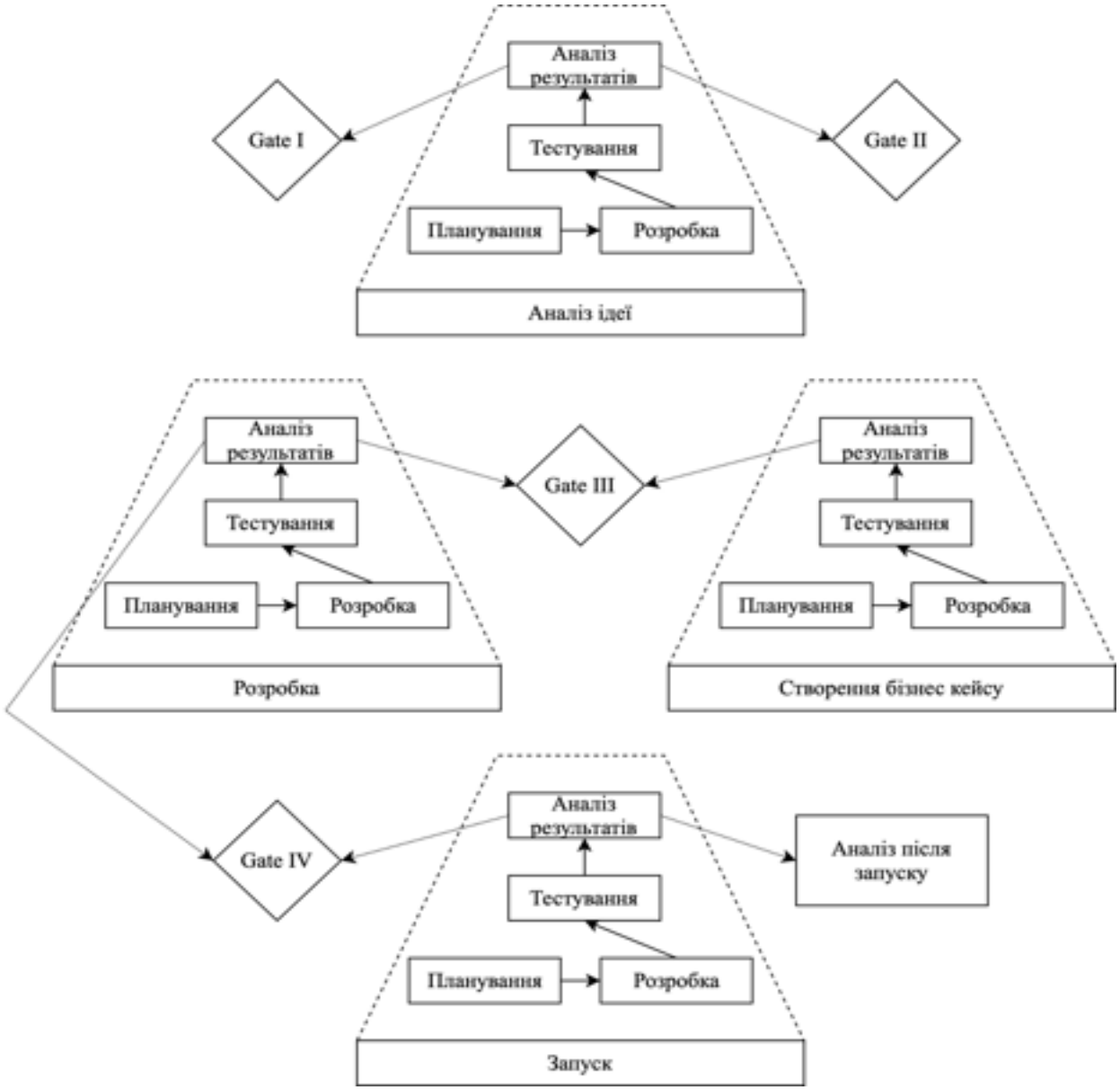


Рисунок 2.18 – Схема Agile Stage-Gate фреймворку

(Джерело – власна розробка)

Прискорений розвиток. Несвоєчасний вихід на ринок є проблемою для багатьох компаній. Ітеративний та швидкий підхід в Agile Stage-Gate фреймворці з його акцентом на розробці робочих прототипів продукту є

перспективним методом вирішення цих часових ризиків. Використання цілеспрямованих команд, які присвячені одному проєкту, також допомагає збільшити швидкість виходу на ринок.

Далі розглянемо шляхи усунення труднощів, пов'язаних з використанням Agile Stage-Gate фреймворку для аутсорс IT-проєктів, продемонструвавши, створену адаптацію фреймворку, враховуючи особливості аутсорс IT-проєктів в цілому та особливості сучасних ітеративних циклів розробки. Також модифікуємо продемонстровану адаптацію для використання нечіткої логіки.

В цілому, компанії по розробці IT-проєктів розподіляються на дві великі категорії: продуктові та аутсорсингові. Продуктові компанії націлені на розробку та підтримку власних продуктів. У аутсорсових компаній більша динаміка розвитку в плані розробки проєктів, адже вона виконує велику кількість різноманітних замовлень одночасно та позбавлена затяжних перед-проєктних та пост-проєктних етапів [28]. Розглянемо особливості розробки аутсорс IT-проєктів детальніше.

Більше можливостей для безперервного стабільного розвитку:

- Продуктові компанії сильно залежать від безперервного потоку ідей нових продуктів для подальшого розвитку. У той же час аутсорс компанії, як правило, отримують готові ідеї для реалізації.
- Якість розробляємого продукту є спільним інтересом для аутсорсових та продуктових компаній. Для продуктових компаній якість впливає на задоволеність кінцевого споживача. У свою чергу для аутсорс компаній якість впливає на задоволеність замовника, що збільшує вірогідність подальшої співпраці та розширення кола замовників [97].
- Прибуток продуктових та аутсорсових компаній відрізняється за характером. Прибуток продуктової компанії є досить динамічним і залежить від маркетингових кампаній вже існуючих та запланованих продуктів. У свою чергу, прибуток аутсорсингової компанії у більшості випадків є різною мірою фіксований і від маркетингу самого продукту не залежить.

Прихильність до змін в процесі розробки: у випадку раптових змін у процесі розробки програмного продукту, таких як зміна специфікацій або функціональності, продуктова компанія стає перед необхідністю виділення додаткового бюджету для підтримки цих змін. Коли аналогічна ситуація постає перед аутсорсинговою компанією, як правило, ці зміні йдуть зі сторони замовника, що, в свою чергу, призводить до того, що матеріальна підтримка цих змін покладається на замовника. Аутсорсова компанія навпаки у виграші, адже розширення етапу розробки тільки збільшує термін співпраці з замовником без додаткових витрат.

Підхід до процесу:

- Для просування своїх продуктів продуктовим компаніям необхідно йти на деякі уступки з метою отримання відгуків від споживачів та розширення споживчої бази, що однозначно призводить до додаткових витрат. Для аутсорсингових компаній таких ситуацій значно менше, а якщо вони і існують, то скоріше за все направлені на підтримку подальшої співпраці з замовником, або на розширення кола замовників.
- В аутсорсингових компаніях значно більше можливостей для роботи над різноманітними проєктами з використанням різноманітних технологій, що робить їх, як правило, дуже привабливими при залученні спеціалістів. З іншого боку, у продуктових компаніях виникає значно більше необхідності в спеціалістах по роботі зі споживачами та клієнтами.
- У продуктових компаніях підхід до випуску версій продукту є досить обачливим, адже майже завжди функціональність випускається у невеликих групах з ціллю спрощення роботи зі споживачами. Це призводить до досить довгого застою на етапі застою та тестування, що у свою чергу призводить до того, що команди по розробці стають мультифункціональними та досить часто перерозподіляються між проєктами.

- Змінюється і підхід до підбору і розподілення персоналу між проєктами. В аутсорсових компаніях більша вірогідність, що технічний спеціаліст або команда технічних спеціалістів є вузькоспеціалізованою та більш кристалізованою, що підвищує якість продукту. В продуктових компаніях часто виникає необхідність зміни технології розробки, адже кожен новий проєкт може вимагати інших підходів до його розробки, що призводить до необхідності деяким технічним спеціалістам постійно змінювати технічну кваліфікацію.

Було проведено аналіз та виявлено загальні етапи для більшості аутсорс проєктів у рамках аутсорсингових компаній:

1. *Пошук проєкту:* ще до початку процесу програмної розробки перед аутсорсинговою компанією стоїть задача пошуку проєкту для розробки. Ця задача є багатофакторною, адже може включати пошук замовника, оцінку ризиків роботи з конкретним існуючим та потенційним замовником та багато інших факторів. Також вона може включати обговорення щодо термінів, технологій та необхідних фінансових і людських ресурсів.
2. *Попередній технічний аналіз проєкту:* після остаточного закріплення відносин між аутсорсинговою компанію і замовником комерційним договором, наступає етап аналізу вимог до проєкту. На відміну від продуктових компаній, аутсорсингові компанії позбавлені необхідності будувати бізнес-кейс та маркетингові стратегії щодо проєкту, що робить процес аналізу технічним. На цьому етапі вже точно обговорені терміни розробки та необхідні для цього ресурси. Основною ціллю цього етапу є розробка попереднього плану процесу програмної розробки та підбір персоналу з наявних або не наявних ресурсів.
3. *Agile спринти програмної розробки:* цей етап є ітераційним, та включає основні процеси, що стосуються програмної розробки. Сучасні Agile технології управління IT-проєктами можуть досить сильно відрізнятися

між собою стосовно процесів, які в них входять. Однак майже з них усіх можна виділити основний скелет процесів:

a. Переоцінка та аналіз спринту;
b. Планування спринту;
c. Програмна розробка функціональності;
d. Тестування функціональності.

4. *Підтримка проєкту:* цей етап є опціональним, адже за контрактною домовленістю з замовником його може і не передбачатися. Якщо його передбачено, то його терміни можуть досить суттєво розпливатися у часі. Це може бути підтримка, пов'язана з первинним запуском продукту або підтримка з завчасно запланованими тривалими термінами.
5. *Ретроспективний аналіз проєкту:* з метою подальшого покращення процесу розробки та розподілення ресурсів між проєктами, проводиться етап аналізу проєкту після його завершення. На основі інформації з попередніх двох етапів та історичних даних (при наявності) формується інформація щодо виконання поточного проєкту з ціллю оновлення історичних даних.

На рис. 2.19 можна детальніше розглянути модифікацію Agile Stage-Gate фреймворку, яка враховує раніше описані особливості аутсорс IT-проєктів з точки зору аутсорсингової компанії.

З рис. 2.19 видно, що у способі розстановки брам між етапами є одна суттєва особливість. Деякі з брам не є однонаправленими.

Наприклад, брама після процесу пошуку проєкту є однонаправленою, адже сам процес пошуку проєкту не є ітераційним. З іншого боку брама перед процесом Agile спринту є двосторонньою, адже після кожного спринту проходить процес оцінки результатів на цій же брамі.

З огляду на попередньо проведений аналіз має сенс розглянути можливість використання нечіткої логіки в Stage-Gate фреймворку.

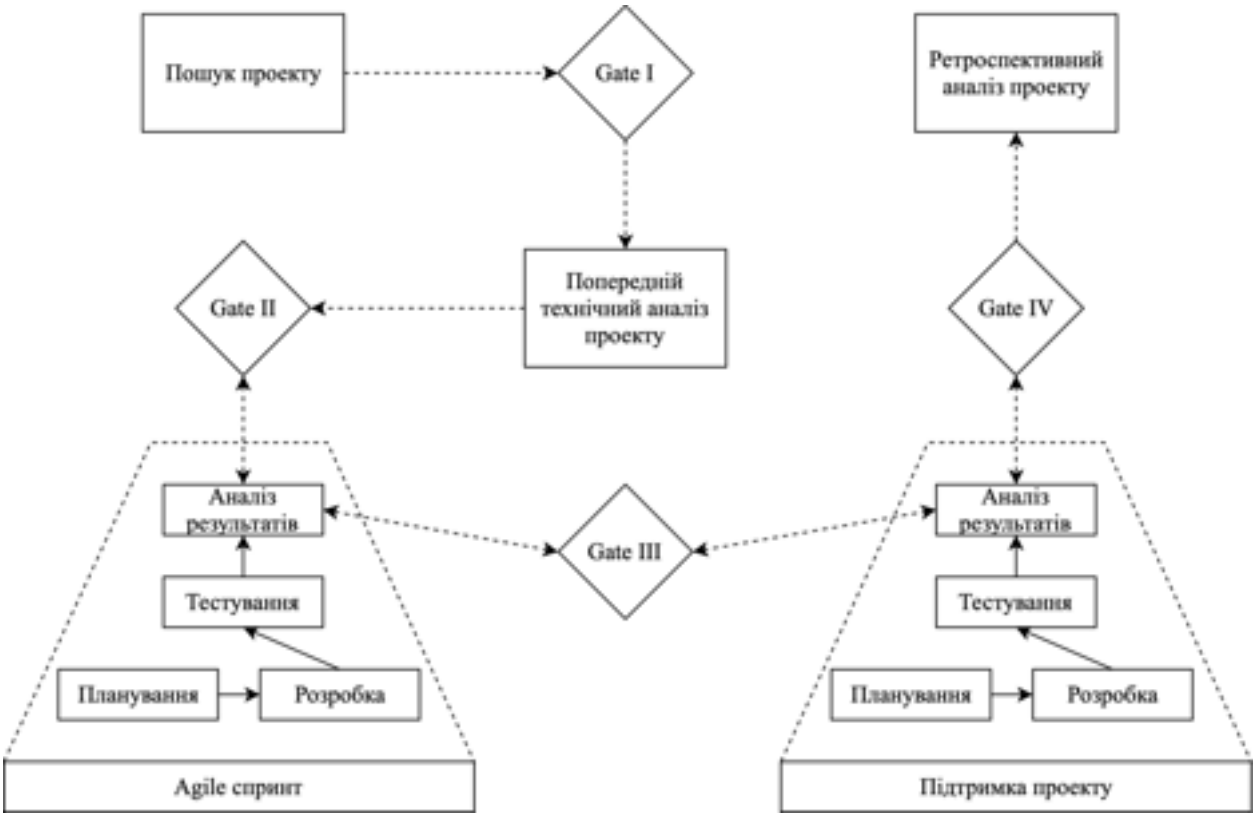


Рисунок 2.19 – Схема модифікації Agile Stage-Gate фреймворку, враховуюча особливості аутсорс IT-проєктів

(Джерело – власна розробка)

Згадуючи попередню аналогію з виробництвом фабричної продукції, спроби використання нечіткої логіки в цьому можуть виглядати штучно, адже на такому виробництві як правило не місце випадковості та неповноті наявної інформації. Але ситуація кардинально змінюється якщо замінити виробництво фабричної продукції на розробку IT-проєкту.

При розробці IT-проєктів є повсякчасною відсутність та нечіткість інформації, а також відсутність повної картини щодо фінального результату. Досить часто результат уточнюється при переході з одного етапу на інший і таким чином кожен наступний етап працює з більш точною інформацією. У зв'язку з цим пропонується модифікація Stage-Gate фреймворку, яка будує нечітку систему для прийняття рішень на кожній границі – рис. 2.20.



Рисунок 2.20 – Stage-Gate фреймворк з використанням нечіткої логіки

(Джерело – власна розробка)

Це надає переваги у порівнянні з класичним Stage-Gate фреймворком: надає можливість описувати правила прийняття рішень та правила переходу з етапу на етап у лексичному форматі; зменшує необхідність у великій кількості точних числових даних у перевагу лексичних даних.

Враховуючи інформацію про переваги використання нечіткої логіки для процесу управління IT-проєктами в цілому, та застосовуючи запропонований підхід для використання нечітких систем для переходу через брами Stage-Gate фреймворку було розроблено загальну структуру модифікації Agile Stage-Gate фреймворку, враховуючи особливості аутсорс IT-проєктів з використанням нечіткої логіки.

Особливістю схеми (рис. 2.21) є використання систем нечіткої логіки при переході від брам до наступних етапів. У блоках «Система нечіткого виводу I…IV» спрацьовує математичний механізм нечіткої логіки, представлений раніше в ході визначення моделі бази знань. Також в деяких випадках є необхідність приміняти системи нечіткої логіки ітеративно в залежності від ітеративності самого етапу. Таком можемо бачити, що у способі розстановки брам між етапами є одна суттєва особливість. Деякі з брам не є однонаправленими.

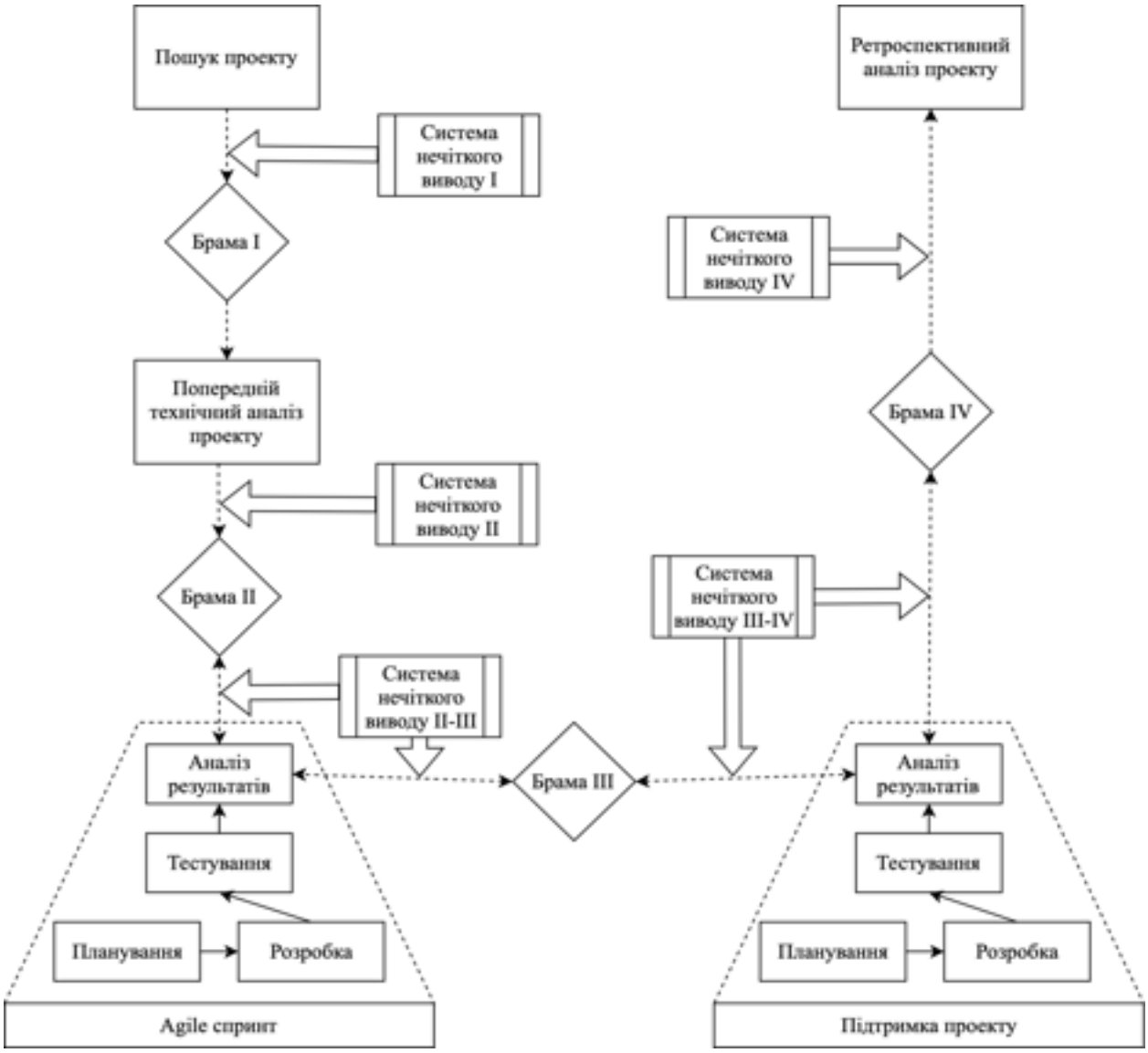


Рисунок 2.21 – Модель Agile Stage-Gate фреймворку з використанням нечіткої логіки

(Джерело – власна розробка)

Наприклад, брама після процесу пошуку проєкту є однонаправленою, адже сам процес пошуку проєкту не є ітераційним. З іншого боку брама перед процесом Agile спринту є двосторонньою, адже після кожного спринту проходить процес оцінки результатів на цій же брамі [43-45].

Висновки до розділу 2

Дослідження, спрямоване на вдосконалення процесу управління аутсорсинговими проєктами, виявило кілька важливих аспектів:

1. Увага до управління аутсорс проєктами з точки зору аутсорсингових компаній є недостатньою, оскільки вона часто зосереджена на питаннях ефективного аутсорсингу для продуктових компаній та інших типів замовників. Цей розрив у розумінні потреб і вимог аутсорсингових компаній може викликати проблеми та перешкоди на шляху досягнення успіху в аутсорсингових проєктах.
2. Виявлено та продемонстровано переваги використання Agile Stage-Gate фреймворку для управління аутсорс проєктами з точки зору аутсорсингових компаній. Цей фреймворк дозволяє гнучко впроваджувати зміни та адаптуватися до змінних вимог клієнтів, забезпечуючи ефективну комунікацію та співпрацю між всіма зацікавленими сторонами. Його використання сприяє покращенню якості та швидкості виконання аутсорсингових проєктів, що є ключовими факторами успіху в цій галузі. Виявлено та продемонстровано переваги використання Agile Stage-Gate фреймворку для процесу управління аутсорс проєктами для аутсорсингових компанії.
3. Розроблена концептуальна модель системи управління аутсорсинговими IT-проєктами, яка базується на використанні нечіткої логіки в загальному контурі прийняття рішень. В основу моделі покладено модифікацію Agile Stage-Gate фреймворку, яка враховує особливості аутсорс IT-проєктів з використанням нечіткої логіки. Ця модифікація дозволяє враховувати не виключно чіткі та однозначні параметри, а також нечіткі та неоднозначні фактори, що є типовими для складних технологічних проєктів. Використання нечіткої логіки дозволяє вирішувати проблеми, коли точна інформація або критерії прийняття рішень не є повністю доступними, що

сприяє підвищенню ефективності та адаптивності процесу управління аутсорсинговими проєктами.

4. Розроблена програмна оболонка нечіткої експертної системи FuzzyKIDE, яка використовує запропоновану оптимальну структуру бази знань та модифікований процес нечіткого виводу. Оболонка забезпечує зручне та ефективне використання нечіткої логіки в процесі управління аутсорсинговими проєктами. Вона допомагає приймати рішення на основі нечітких даних та неоднозначних ситуацій, що сприяє зниженню ризику та підвищенню якості управління.

Загалом, дослідження вказує на важливість розуміння та врахування особливостей управління аутсорсинговими проєктами з точки зору аутсорсингових компаній. Використання Agile Stage-Gate фреймворку та нечіткої логіки може забезпечити ефективність, гнучкість та адаптивність управління, а програмна оболонка FuzzyKIDE допоможе покращити процес розрахунків та прийняття рішень на основі нечітких даних. Ці підходи в сукупності можуть сприяти покращенню якості та успішності управління аутсорсинговими проєктами з точки зору аутсорсингових компаній.

Результати розділу 2 висвітлено у таких працях автора, наведених у Додатку А: [1, 2, 3, 4, 7, 10, 11].

РОЗДІЛ 3
ЕКСПЕРТНІ КОНСУЛЬТАЦІЇ З FUZZYKIDE ЯК АНАЛІТИЧНА ОСНОВА ОБГРУНТУВАННЯ ТА ПРИЙНЯТТЯ ЕКОНОМІЧНИХ РІШЕНЬ З УПРАВЛІННЯ ПОРТФОЛІО ПРОЄКТІВ

3.1 Розв’язання задач стратегічного управління ІТ-компанією в умовах невизначеного середовища

Розглянемо приклад використання програмної оболонки, а саме розглянемо задачу допомоги в прийнятті рішень відповідно до стратегічних цілей аутсорсингової ІТ-компанії. Відповідно до розділу 2.4, ця задача не обмежена окремими етапами циклу розробки програмного забезпечення в аутсорсинговій ІТ-компанії і може виникнути на кожному з етапів циклу в залежності від конкретної стратегічної цілі.

Досить часто менеджери в аутсорсингових компаніях приймають рішення відповідно до поточних стратегічних цілей компанії. Стратегічні цілі розбиваються на чотири блоки:

- Фінанси.
- Клієнти.
- Бізнес-процеси.
- Ріст та навчання.

У кожному з блоків існує дуже багато різноманітних стратегічних цілей, які можуть стояти перед аутсорсинговою ІТ-компанією. Перечислимо деякі з них. Прикладом стратегічної цілі в блоці «Фінанси» може бути зростання прибутку. Прикладами стратегічних цілей в блоці «Клієнти» можуть бути: своєчасно пропонувати продукти та послуги з високою якістю та низькою ціною; збільшення клієнтської бази. Для досягнення стратегічних цілей, зазначених в блоці «Клієнти», необхідно розглянути стратегічні цілі в блоці

«Бізнес-процеси»: зниження витрат на розробку програмного забезпечення; підвищення ефективності та якості розробки програмного забезпечення; спеціалізація на конкретних типах проєктів розробки програмного забезпечення.

Після того, як всі стратегічні цілі сформовані, менеджер будує план роботи, розподіляючи стратегічну ціль на тактичні цілі (завдання) своїх підлеглих. Таким чином, допомога в прийнятті рішень в процесі слідування стратегічній цілі є критичною в процесі роботи як менеджера, так і його підлеглих. Далі розглянемо декілька прикладів допомоги в прийнятті рішень в слідуванні стратегічній цілі.

В процесі розвитку аутсорсингових IT-компаній майже завжди виникає необхідність обрати шлях для професійного спрямування компанії. Наприклад, на початковій стадії розвитку компанії існує дуже мало можливостей для обрання проєктів бажаної спрямованості, тому обираються проєкти з найменших ризиком та найбільшим прибутком незалежно від типу проєкту. Через деякий час у компанії з'являється можливість відмовлятися від конкретних проєктів в перевагу проєктам, що відповідають бажаному професійному спрямуванню.

3.1.1 Стратегічна орієнтація компанії: професійне спрямування

Розглянемо модельну ситуацію розвитку аутсорсингової IT-компанії, ціллю якої є *професійне спрямування як компанії, що спеціалізується на розробці проєктів телекомунікаційного (ТК) типу*. Розвиток компанії буде розділено на 4 етапи і на кожному етапі буде надано рекомендацію прийняття рішення щодо обрання проєктів в розробку. Будемо мати на увазі, що поточна модельна ситуація на ринку можливих проєктів не буде сильно відрізнятися залежно від етапу. Натомість відрізнятися буде поточний модельний стан

аутсорсингової ІТ-компанії, а саме уже наявні проєкти, зайнятість доступних команд розробки, тощо. Скорочений опис етапів:

- Починаюча компанія з можливістю обрати ТК проєкт;
- Все ще починаюча компанія, але без можливості обрати ТК проєкт;
- Компанія середнього розвитку з можливістю обрати ТК проєкт;
- Розвинута компанія з можливістю обрати ТК проєкт.

Детальніший опис етапів буде надано далі в рамках кожної окремої консультації. Поки що розглянемо шаблони правил, необхідних для побудови бази знань для вирішення цієї задачі.

Перший набір правил формується на основі інформації, що дозволяє оцінити пріоритетність проєктів, базуючись на поточних проєктах. В залежності від кількості проєктів того чи іншого типу, можливо оцінити якому типу проєктів слід віддати пріоритет. Цей набір правил є досить скромним, але створює необхідну інфраструктуру для оцінки можливості взяти окремо розглядаємий проєкт у розробку.

Сформулюємо шаблони для цих правил:

- *ЯКЩО (Кількість ТК проєктів = N1*
  ТА Кількість не ТК проєктів = N2)
  ТО (Пріоритет ТК проєктів = N3
  ТА Пріоритет не ТК проєктів = N4)
- *ЯКЩО (Пріоритет ТК проєктів = N1*
  ТА Пріоритет не ТК проєктів = N2)
  ТО (Пріоритетний тип проєкту = {ТК, не ТК})
- *Тощо.*

Наступний набір правил формується на основі наявної інформації щодо потенційних замовників. Детальність шаблонів може залежати від ступеню деталізації інформації про замовників, але фінальним результатом є оцінка ризику взаємодії з замовником, що дозволяє уточнити інформацію щодо пріоритету окремих типів проєктів. Далі сформулюємо шаблони для цих правил:

- *ЯКЩО (Ризик взаємодії з замовником X = {Низький, Середній, Високий}*
  ТА Проєкт Y має {ТК, не ТК} тип
  ТА Пріоритетний тип проєкту = {ТК, не ТК})
  ТО ({Є, немає} рекомендації взяти проєкт Y у розробку)
- *ЯКЩО (Один з {ТК, не ТК} проєктів є рекомендація взяти у розробку)*
  ТО (Є можливість взяти у розробку {ТК, не ТК} проєкт)
- *Тощо.*

Цей набір правил теж може видатися досить скромним, але ці шаблони дозволяють згенерувати велику кількість правил навіть при мінімальних варіаціях в типах проєктів. Також ці шаблони можуть бути розширені у разі наявності більш детальної інформації щодо оцінки ризику роботи з замовником.

Хоча у попередньому наборі правил можемо бачити шаблон щодо рекомендації взяття проєкт в розробку, дуже важливою частиною цієї задачі є оцінка приблизного поточного статусу уже наявних проєктів. Це може допомогти у рішенні щодо необхідності розпочати процес рекрутингу у разі нестачі команд необхідної кваліфікації у момент, коли планується розпочати новий проєкт того чи іншого типу. Розглянемо шаблони для цих правил:

- *ЯКЩО (Поточний проєкт X завершиться {Скоро, Не скоро}*
  ТА Проєкт X має тип {ТК, не ТК})
  ТО ({Скоро, Не скоро} звільниться команда проєкту X типу {ТК, не ТК})
- *ЯКЩО ({Скоро, Не скоро} звільниться одна з команд {ТК, не ТК})*
  ТО (У наявності {Є, Немає} команди {ТК, не ТК})
- *ЯКЩО (Є можливість взяти у розробку {ТК, не ТК} проєкт*
  ТА У наявності {Є, Немає} команди {ТК, не ТК} типу)
  ТО ({Є, Немає} необхідності в пошуку {ТК, не ТК} спеціалістів)
- *ЯКЩО (Є необхідність в пошуку ТК спеціалістів*
  АБО Є необхідність в пошуку не ТК спеціалістів)
  ТО (Є необхідність почати процес рекрутингу)
- *Тощо.*

Змоделюємо тестову ситуацію для демонстрації рішення задачі стратегічної цілі професійного спрямування аутсорсингової ІТ-компанії. Розглянемо компанію впродовж чотирьох етапів розвитку. Кожен етап відрізняється ступенем поточного розвитку компанії, наявних у розробці проєктами. Також кожен етап відрізняється поточним станом ринку проєктів, а саме проєктів доступних для розгляду щодо взяття у розробку. Розглянемо наявну інформацію щодо кожного етапу у табл. 3.1 – 3.4.

Почнемо з етапу, коли компанія на початковому рівні розвитку. Цей етап характеризується низькою кількістю початкових проєктів, що робить пріоритети ТК та не ТК проєктів однаковими, та наявністю ТК проєктів на ринку – табл. 3.1. Наступний етап також є початковим, але відрізняється від попереднього відсутністю ТК проєктів на ринку – табл. 3.2. Наступний етап характеризує компанію середнього рівня розвитку, що дозволяє підняти пріоритет ТК проєктів у порівнянні з не ТК проєктами, та наявністю ТК проєктів на ринку – табл. 3.3. Останній етап характеризує розвинуту компанію, яка може дозволити собі ігнорувати не ТК проєкти. Також на ринку доступні ТК проєкти – табл. 3.4.

Таблиця 3.1 – Поточна проєктна ситуація – фаза 1 (Джерело – власна розробка)

Назва	**Тип**	**Компанія**	**Ризик співпраці**	**Завершення**
1	2	3	4	5
CycleDuo	ТК	Виконавець	-	Скоро
Hymera	Не ТК	Виконавець	-	Не скоро
Templater	Не ТК	Виконавець	-	Скоро
Exploration	ТК	Global State	Низький	-
Genesis	ТК	Mazzle	Високий	-
Crowding	ТК	Mazzle	Високий	-
Firantis	Не ТК	Global State	Низький	-
Fabulous	Не ТК	Global State	Низький	-

Базуючись на початковій інформації у цьому розділі та на прикладах побудови правил, описаних раніше, формується набір нечітких правил бази

знань для рішення задачі пошуку проєктів стосовно стратегічної цілі. Також, на основі інформації у цьому розділі формуються початкові дані, необхідні для вирішення цієї задачі.

Перейдемо до демонстрації рішення задачі пошуку проєктів стосовно стратегічної цілі аутсорсингової IT-компанії. На основі попередньо продемонстрованих шаблонів правил, додаємо новий профіль до програмної оболонки та заповнимо базу знань – рис. 3.1. На основі інформації щодо поточних статусів проєктів (табл. 3.1), сформуємо початкові дані та почнемо процес виводу.

Таблиця 3.2 – Поточна проєктна ситуація – фаза 2 (Джерело – власна розробка)

Назва	**Тип**	**Компанія**	**Ризик співпраці**	**Завершення**
1	2	3	4	5
CycleDuo	ТК	Виконавець	-	Не скоро
Hymera	Не ТК	Виконавець	-	Не скоро
Templater	Не ТК	Виконавець	-	Не скоро
Exploration	Не ТК	Global State	Низький	-
Genesis	Не ТК	Mazzle	Високий	-
Crowding	Не ТК	Mazzle	Високий	-
Firantis	Не ТК	Global State	Низький	-
Fabulous	Не ТК	Global State	Низький	-

Таблиця 3.3 – Поточна проєктна ситуація – фаза 3 (Джерело – власна розробка)

Назва	**Тип**	**Компанія**	**Ризик співпраці**	**Завершення**
1	2	3	4	5
CycleDuo	ТК	Виконавець	-	Не скоро
Avion	ТК	Виконавець	-	Не скоро
Howl	ТК	Виконавець	-	Не скоро
Converge	ТК	Виконавець	-	Не скоро
Hymera	Не ТК	Виконавець	-	Не скоро
Templater	Не ТК	Виконавець	-	Скоро
Exploration	Не ТК	Global State	Низький	-
Genesis	ТК	Mazzle	Високий	-
Crowding	Не ТК	Mazzle	Високий	-
Firantis	Не ТК	Global State	Низький	-
Fabulous	Не ТК	Global State	Низький	-

Таблиця 3.4 – Поточна проєктна ситуація – фаза 4 (Джерело – власна розробка)

Назва	Тип	Компанія	Ризик співпраці	Завершення
1	2	3	4	5
CycleDuo	ТК	Виконавець	-	Скоро
Avion	ТК	Виконавець	-	Не скоро
Howl	ТК	Виконавець	-	Не скоро
Converge	ТК	Виконавець	-	Не скоро
Harlan	ТК	Виконавець	-	Не скоро
Hyperion	ТК	Виконавець	-	Не скоро
Hymera	Не ТК	Виконавець	-	Не скоро
Templater	Не ТК	Виконавець	-	Скоро
Exploration	Не ТК	Global State	Низький	-
Genesis	ТК	Mazzle	Високий	-
Crowding	Не ТК	Mazzle	Високий	-
Firantis	Не ТК	Global State	Низький	-
Fabulous	Не ТК	GlobalState	Низький	-

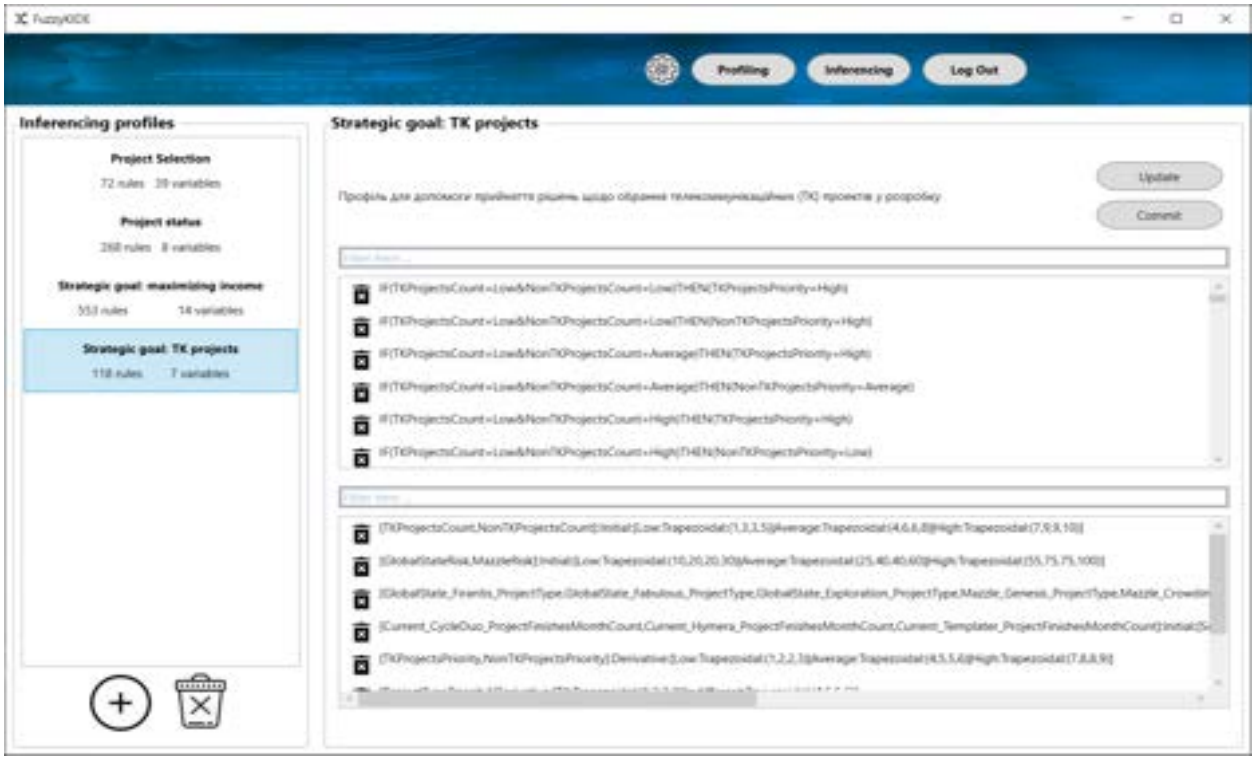


Рисунок 3.1 – Заповнена база знань для задачі пошуку проєкту відносно стратегічної цілі (Джерело – власна розробка)

На рис. 3.2 можемо побачити результат нечіткого виводу щодо взяття проєкту Exploration у розробку: *Global State Exploration Possible = Yes (60) was enabled with confidence factor 0.7*.

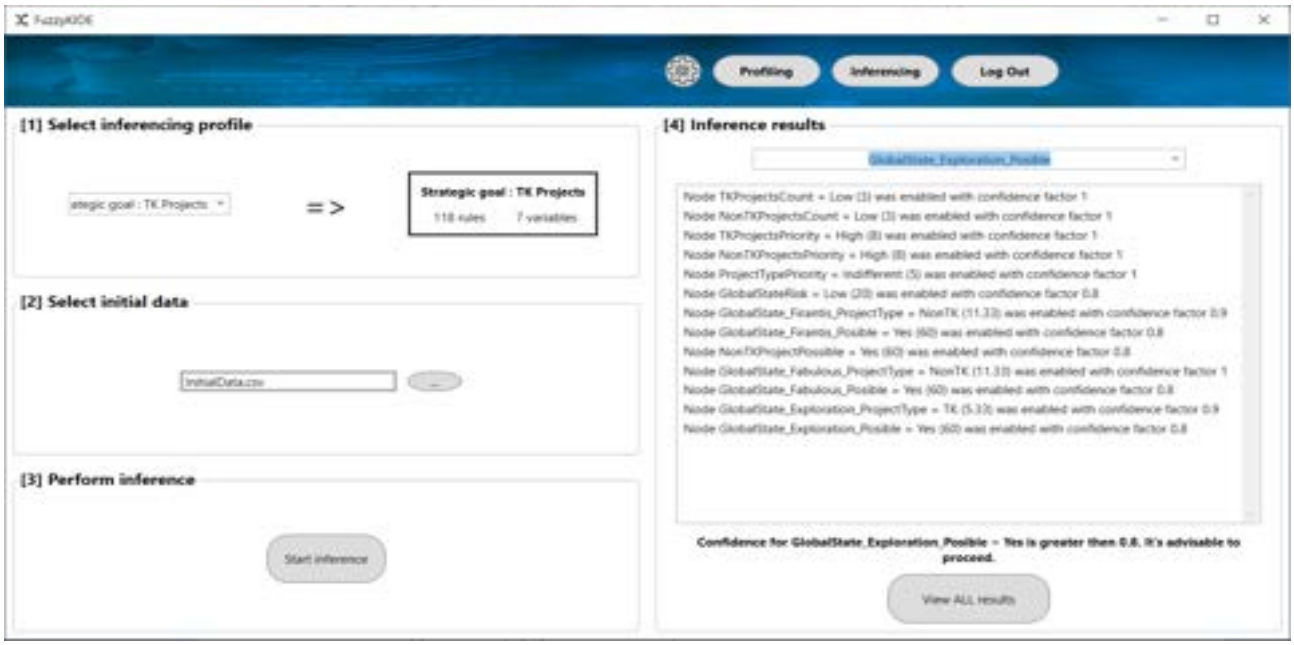


Рисунок 3.2 – Рішення задачі пошуку проєкту відносно стратегічної цілі: проєкт (Джерело – власна розробка)

Вузол *Global State Exploration Possible = Yes (60)* було активовано з коефіцієнтом достовірності 0.8, що дорівнює пороговому значенню за замовчуванням. Це свідчить про те, що експертна система рекомендує взяти у розробку проєкт Exploration замовника Global State. Це є логічним, адже це проєкт ТК типу від замовника з низьким ризиком співпраці. Нечіткому значенню *Yes* відповідає чітке значення 60.

Результат виводу є багатофакторним та з огляду на інші проєкти можемо бачити, що усі вузли доступних ТК проєктів було розраховано у значення *Yes*, але рекомендуються дії тільки для проєкту Exploration. Це досягається за рахунок того, що ризик взаємодії з компанією Mazzle є досить вагомим. Якщо користувач вважатиме цей ризик виправданим, то у користувача є можливість зменшити порогове значення коефіцієнту достовірності та повторити процес нечіткого виводу.

На рис. 3.3 та 3.4 можемо побачити результат нечіткого виводу щодо рекомендацій рекрутингу:

- *StartRecrutingProcess = No (16.67) was enabled with confidence factor 0.7;*
- *TKStartRecrutingProcess = No (16.67) was enabled with confidence factor 0.8*.

Вузол *StartRecrutingProcess = No (16.67)* було активовано з коефіцієнтом достовірності 0.7, що нижче порогового значення за замовчуванням (0.8). Це означає, що експертна система не впевнена щодо відсутності необхідності починати процес рекрутингу в компанію. Якщо проаналізувати процес виводу, то можемо побачити, що система не впевнена через невпевненість, що команда не ТК проєкту Templater встигне завершити свій проєкт вчасно.

З другого боку вузол *TKStartRecrutingProcess = No (16.67)* було активовано з коефіцієнтом достовірності 0.8, що дорівнює пороговому значенню за замовчуванням (0.8).



Рисунок 3.3 – Рішення задачі пошуку проєкту відносно стратегічної цілі: загальний рекрутинг (Джерело – власна розробка)

Це означає, що експертна система впевнена щодо можливості ТК команди CycleDuo завершити проєкт вчасно. У обох результатах нечіткому значенню *No* відповідає чітке значення 16.67. Взагалом можемо зробити

висновок, що система не бачить необхідності у пошуку нової ТК команди, але не виключає можливості пошуку нової команди для не ТК проєктів.

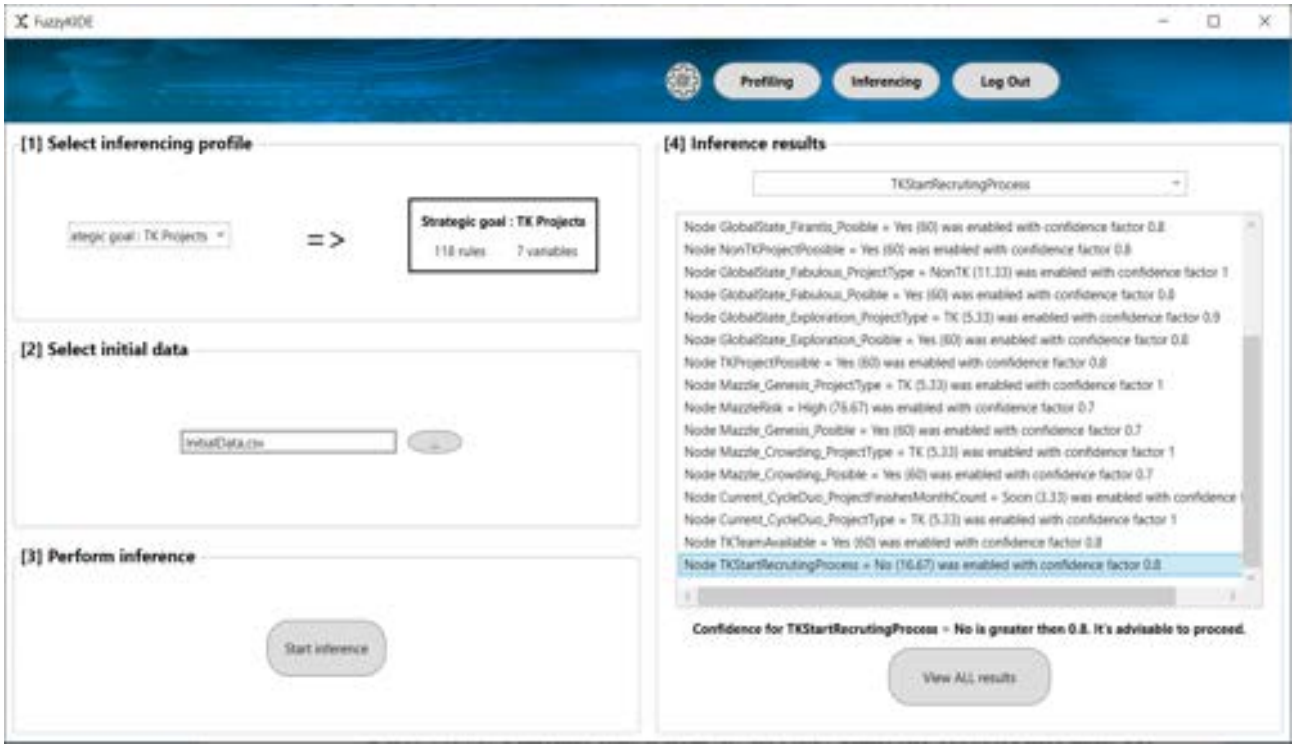


Рисунок 3.4 – Рішення задачі пошуку проєкту відносно стратегічної цілі: ТК рекрутинг (Джерело – власна розробка)

На рис. 3.5 можемо бачити один з результатів нечіткого виводу щодо обрання проєкту у розробку у другій фазі: *Mazzle Genesis Possible = No (16.67) was enabled with confidence factor 0.7*.

Друга фаза відрізняється від першої відсутністю ТК проєктів на ринку, що сильно впливає на процес виводу. Результат виводу є багатофакторним і активування вузла *Mazzle Genesis Possible = No (16.67)* є лише одним з прикладів виводу.

У загалом, адже усі наявні на ринку не ТК проєкти мають однаковий пріоритет, система рекомендує брати у розробку проєкти компанії Global State через низький ризик співпраці з замовником та навпаки не рекомендує брати у розробку проєкти компанії Mazzle.

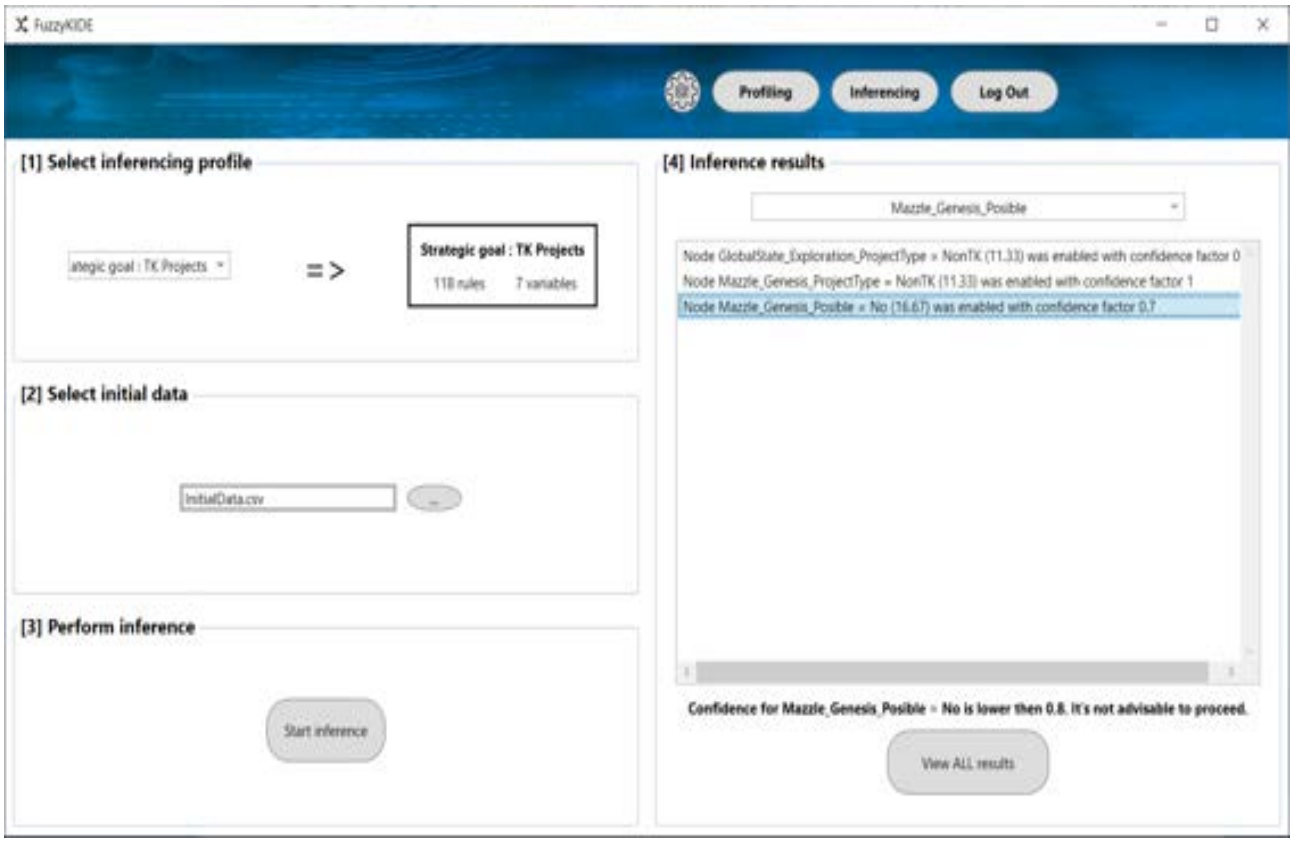


Рисунок 3.5 – Рішення задачі пошуку проєкту відносно стратегічної цілі: проєкт (Джерело – власна розробка)

На рис. 3.6 можемо побачити результат нечіткого виводу щодо рекомендацій рекрутингу: *NonTKStartRecrutingProcess = Yes (60) was enabled with confidence factor 0.7*. Вузол *NonTKStartRecrutingProcess = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що нижче порогового значення за замовчуванням (0.8).

Це означає, що експертна система на основі наявної інформації не впевнена, що у найближчий час не звільниться ні одна команда та, у свою чергу, не впевнена чи варто починати процес пошуку нової не ТК команди. Нечіткому значенню *Yes* відповідає чітке значення 60.

На рис. 3.7 можемо бачити результат нечіткого виводу щодо обрання проєкту у розробку у третій фазі: *Mazzle Genesis Possible = Yes (60) was enabled with confidence factor 0.7*.

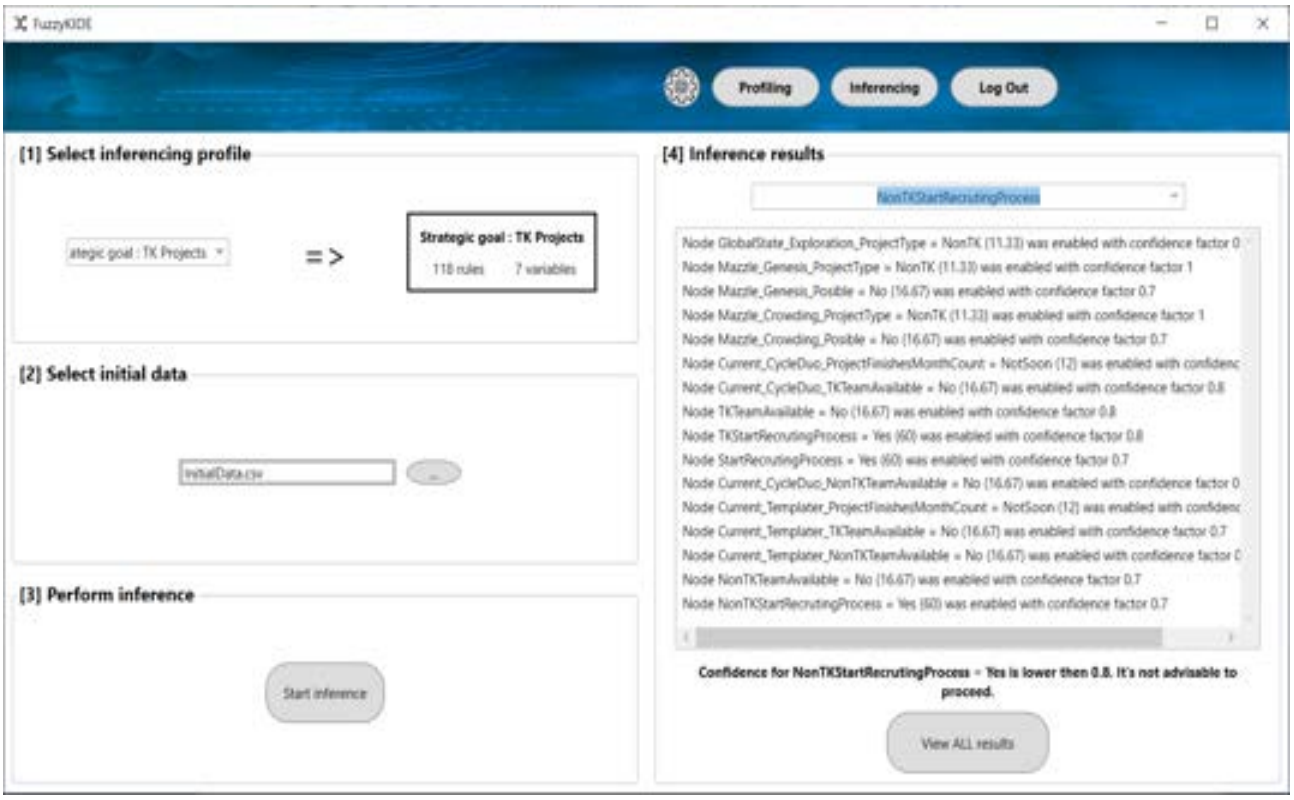


Рисунок 3.6 – Рішення задачі пошуку проєкту відносно стратегічної цілі: не ТК рекрутинг (Джерело – власна розробка)

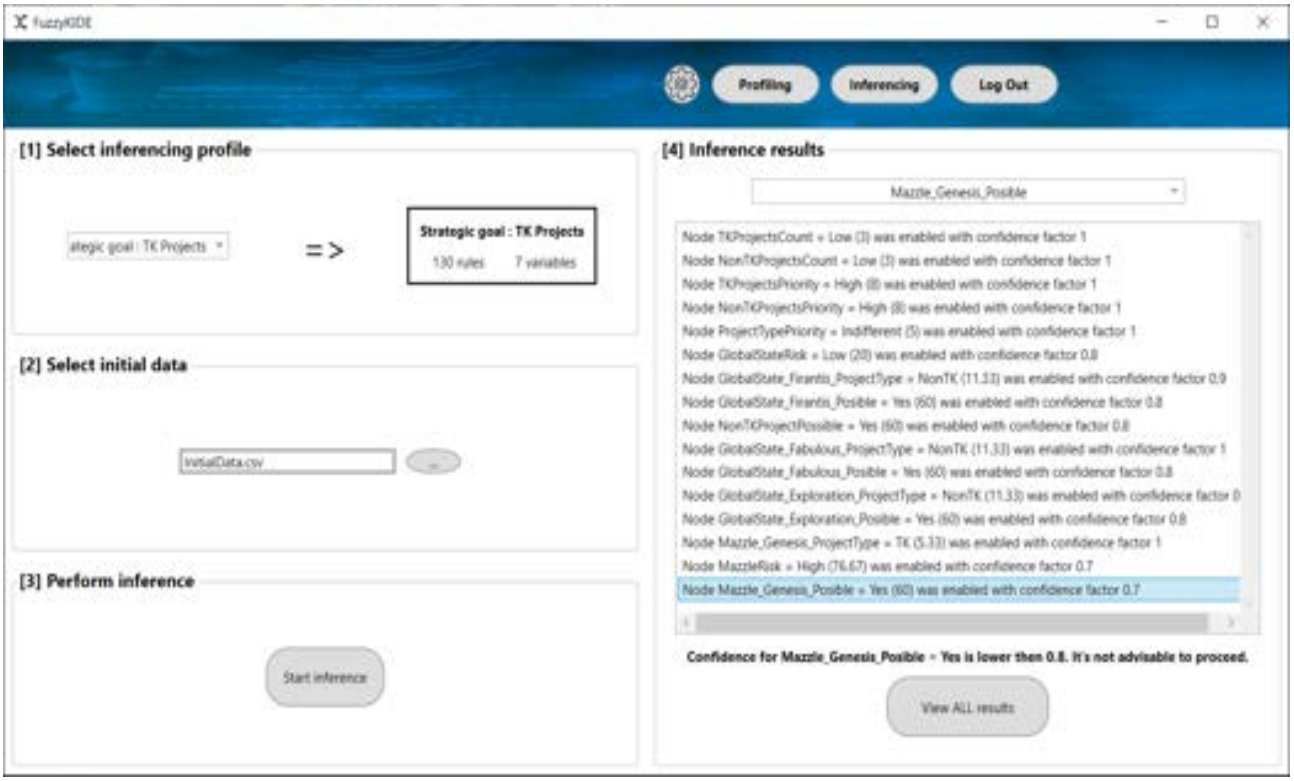


Рисунок 3.7 – Рішення задачі пошуку проєкту відносно стратегічної цілі: проєкт (Джерело – власна розробка)

Вузол *Mazzle Genesis Possible = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що нижче порогового значення за замовчуванням (0.8). Це означає, що експертна система не впевнена чи дійсно варто брати проєкт Genesis у розробку через високий ризик взаємодії з компанією Mazzle. З іншої сторони, якщо розглянути не ТК проєкт компанії Mazzle, наприклад проєкт Crowding, то можемо бачити, що активується вузол *Mazzle Crowding Possible = Yes* з коефіцієнтом достовірності вищим за порогове значення, що свідчить про однозначну рекомендацію не брати проєкт Crowding у розробку через комбінацію того, що це не ТК проєкт та високий ризик роботи з замовником.

На рис. 3.8 можемо побачити результат нечіткого виводу щодо рекомендацій рекрутингу: *StartRecrutingProcess = Yes (60) was enabled with confidence factor 0.7*.

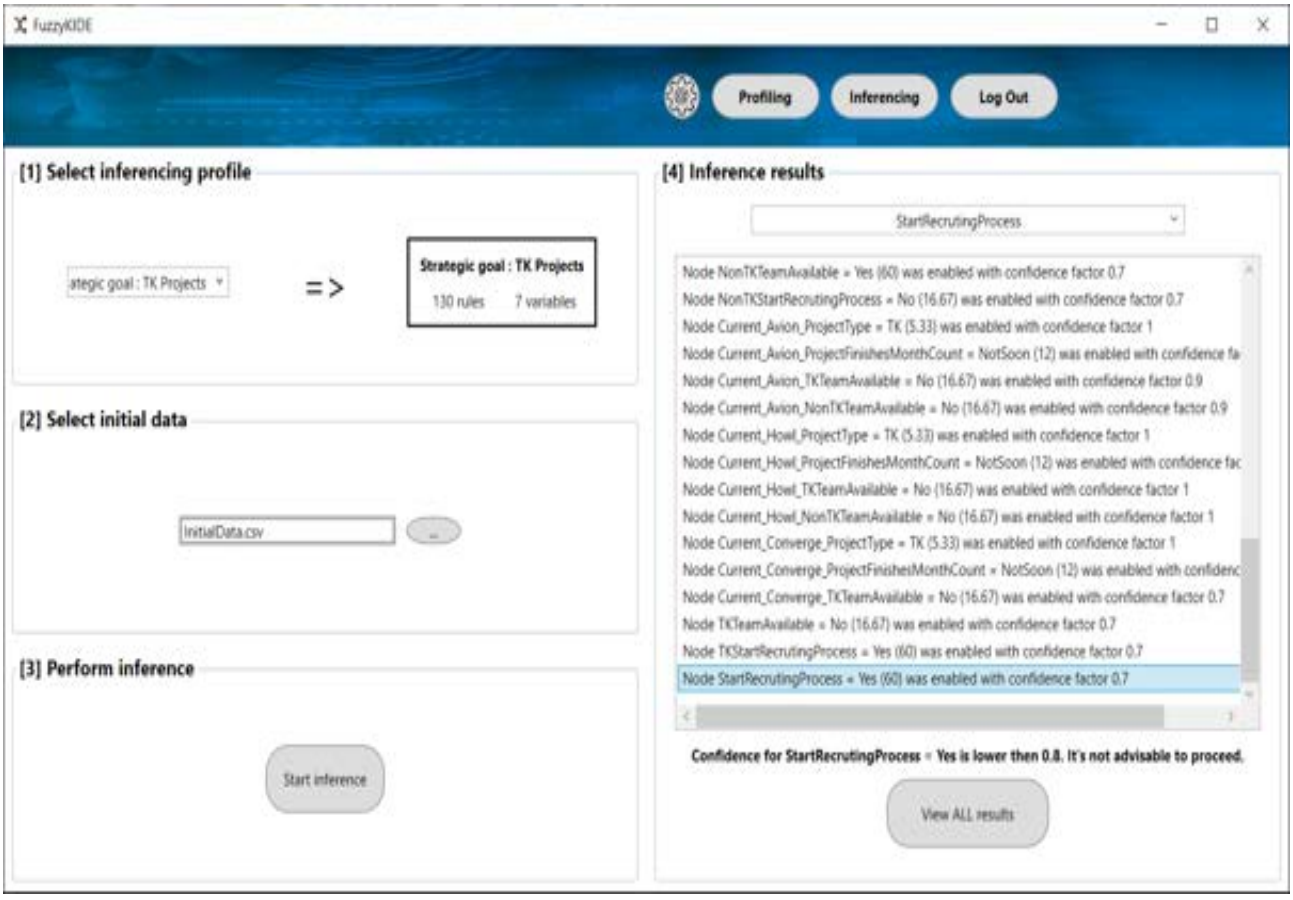


Рисунок 3.8 – Рішення задачі пошуку проєкту відносно стратегічної цілі: загальний рекрутинг (Джерело – власна розробка)

Вузол *StartRecrutingProcess = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що нижче порогового значення за замовчуванням (0.8).

Це означає, що експертна система не впевнена щодо рекомендації почати процес рекрутингу, тому що не впевнена, що у найближчий час звільниться команда. Якщо розглянути процес виводу детально, то можна побачити, що річ іде про доступність ТК команди. У обох результатах нечіткому значенню *Yes* відповідає чітке значення 60.

На рис. 3.9 можемо бачити результат нечіткого виводу щодо обрання проєкту у розробку у четвертій фазі: *Mazzle Genesis Possible = Yes (60) was enabled with confidence factor 0.7*.

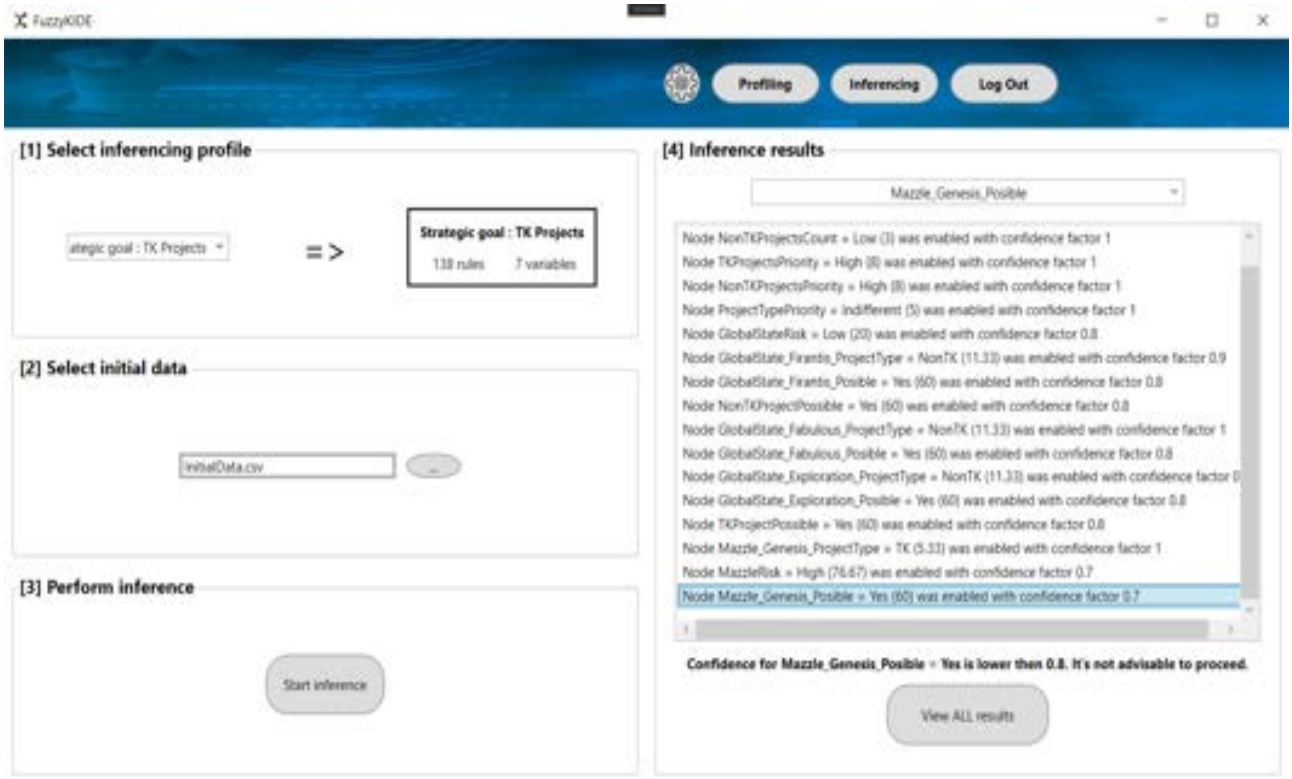


Рисунок 3.9 – Рішення задачі пошуку проєкту відносно стратегічної цілі: проєкт (Джерело – власна розробка)

Вузол *Mazzle Genesis Possible = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що нижче порогового значення за замовчуванням (0.8). Цей результат схожий за суттю аналогічного результату третьої фази. Нечіткому значенню *Yes* відповідає чітке значення 60. Розглянемо частину, яка відрізняє результати виводу цих фаз, а саме рекомендації щодо рекрутингу.

На рис. 3.10 можемо побачити результат нечіткого виводу щодо рекомендацій рекрутингу: *TKStartRecrutingProcess = No (16.67) was enabled with confidence factor 0.8*.

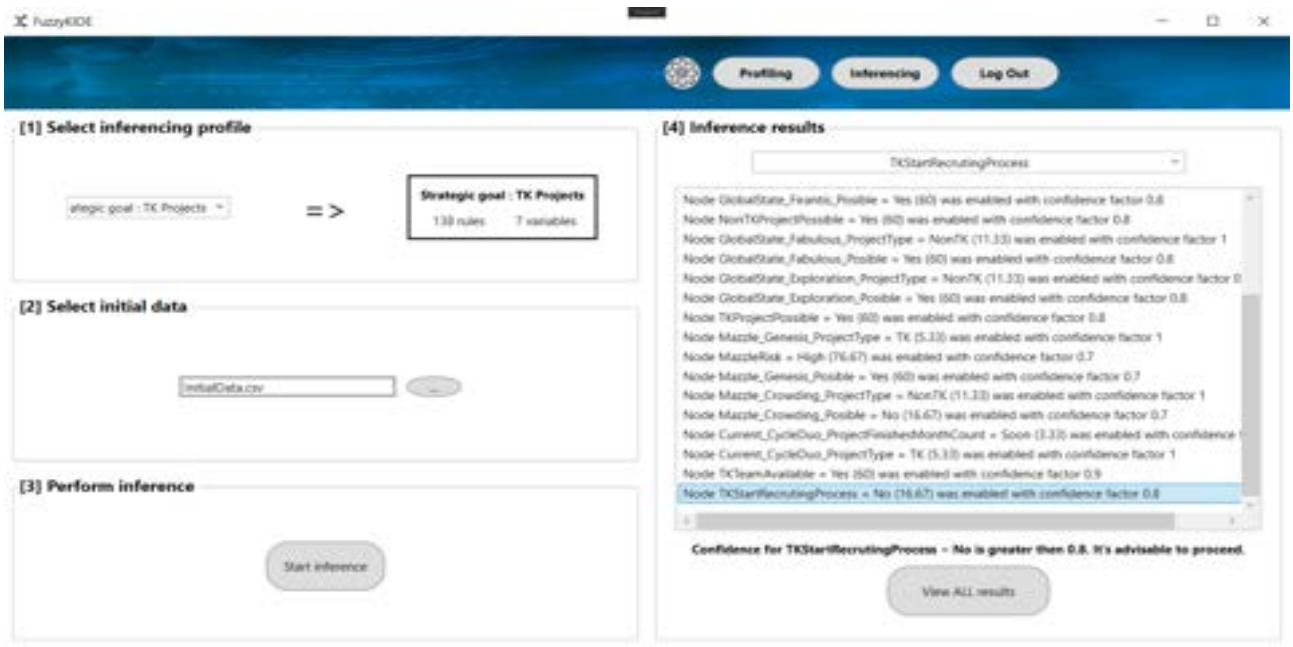


Рисунок 3.10 – Рішення задачі пошуку проєкту відносно стратегічної цілі: ТК рекрутинг (Джерело – власна розробка)

Вузол *TKStartRecrutingProcess = No (16.67)* було активовано з коефіцієнтом достовірності 0.8, що дорівнює пороговому значенню за замовчуванням (0.8). Це означає, що експертна система впевнена щодо рекомендації не починати процес пошуку нової ТК команди, тому що не впевнена, що у найближчий час звільниться одна з наявних ТК команд. Нечіткому значенню *No* відповідає чітке значення 16.67.

3.1.2 Стратегічна орієнтація компанії: максимізація прибутку

Ціллю кожної аутсорсингової IT-компанії є максимізація прибутку за, свого роду, послуги посередництва між компанією замовником та командами

розробки. Чим вдаліше було обрано команду для замовника, тим більша ймовірність успішного завершення проєкту та подальшої співпраці з замовником. Але треба мати на увазі, що основної ціллю в цьому процесі є мінімізація витрат. Наприклад, підбір команд з найменшим ризиком невиконання проєкту та найбільшою різницею в прибутку від проєкту замовника та компенсацією за роботи команди. Це дає можливість збереження коштів на підтримку адміністративного персоналу, розробку маркетингових компаній та інші активності.

Також з ціллю максимізації прибутку аутсорсинговій компанії необхідно звертати увагу на витрати зі сторони адміністративного персоналу, витрати на різноманітні рекламні компанії, витрати на підтримку активного офісу, тощо. Необхідно зважити ризики підтримки цих витрати та співвіднести їх до користі, яку вони приносять. Дуже часто є можливість прийняти рішення щодо відмови або скорочення витрат на активності, які себе не виправдовують, що позитивно вплине на прибуток компанії.

Розглянемо модельну ситуацію аутсорсингової IT-компанії, ціллю якої є максимізація прибутку в наступних умовах:

- Є набір наявних проєктів у розробці;
- Є набір проєктів на ринку для розгляду, щодо обрання у розробку;
- Є інформація щодо витрати та корисності різноманітних рекламних кампаній;
- Є інформація щодо розміру компанії, розміру та витрат на адміністративний персонал та витрат на підтримку активного офісу.

Детальніший опис цих умов буде надано далі в рамках кожної окремої консультації та під час формування бази знань. Поки що розглянемо шаблони правил, необхідних для побудови бази знань для вирішення цієї задачі.

Перший набір правил формується на основі наявної інформації про потенційних замовників та їх проєкти, а саме ризик співпраці с замовником, прибуток від проєктів та тип цього прибутку. Важлива інформація про потенційні нові проєкти існуючих замовників, адже це дає можливість оцінити

рівень відходження від плануємих оцінок. Так, наприклад, якщо часові оцінки виконання проєктів уже існуючого замовника занижені, то можливо є сенс підняти пріоритет проєктів, адже це свідчить про більший рівень прибутку ніж плануємий. Також аналогічна ситуація стосується типу прибутку від проєктів. Проєкти які виконуються за моделлю фіксованої ціни, як правило, мають менший пріоритет, адже у разі виконання проєкту довше запланованого, рівень компенсації від замовника не змінюється, що може негативно вплинути на рівень прибутку. Зважаючи на цю інформацію можемо сформувати наступні шаблони правил:

- *ЯКЩО (Ризик співпраці з компанією X = N1*
  ТА Прибуток проєкту Y від компанії X = N2)
  ТО ({Є, Немає} сенсу розглядати проєкт Y від компанії X)
- *ЯКЩО (Компанія X є поточним замовником*
  ТА Є сенс розглядати проєкт Y від компанії Y ТА Відхилення в оцінках компанії X = N1
  ТА Тип прибутку проєкту Y від компанії X = N2)
  ТО (Пріоритет проєкту Y = N3)
- *ЯКЩО (Немає сенсу розглядати проєкт Y від компанії X)*
  ТО (Пріоритет проєкту Y відсутній)
- *Тощо.*

Наступний набір правил допомагає оцінити рівень ризику доручення проєкту уже наявній команді розробки та сформувати пріоритет команди в залежності від її поточного рівня компенсації та ймовірного часу завершення поточного проєкту. Команди, які не скоро закінчать поточний проєкт немає сенсу розглядати в процесі пріоритезації. Також команди з низьким рівнем компенсації та низьким рівнем ризику співпраці отримують вищий пріоритет в процесі співвідношення до потенційних проєктів, адже це дозволить збільшити різницю між компенсацією замовника та команди, що, у свою чергу, збільшить прибуток компанії. Зважаючи на цю інформацію можемо сформувати наступні шаблони правил:

- *ЯКЩО (Команда X скоро завершує проєкт*
  ТА Компенсація команди X = N1
  ТА Ризик взаємодії з командою X = N2)
  ТО (Пріоритет команди X = N3)
- *ЯКЩО (Команда X не скоро завершує проєкт)*
  ТО (Пріоритет команди X відсутній)
- *Тощо.*

Цей набір правил збирає інформацію виводу з попередніх наборів та формує можливі пари відповідностей потенційних проєктів замовників та наявних команд, які скоро завершають свої поточні проєкти. Цей набір правил є найбільш нечітким серед запропонованим, адже на нього найбільше впливає досвід експерта, який його формує. Досить складно сформувати єдину думку щодо відповідності пріоритетів команд та проєктів, адже існує ймовірність, що зі зміною правил з попередніх наборів, наприклад у ситуації їх розширення іншим експертом, також може змінитися і цей набір правил. Але у цьому наборі правил є і незмінна частина – пропозиції щодо набору нових команд або скорочення існуючих у разі неможливості знайти відповідність. Так, дуже частою є практика розпуску команд для яких впродовж деякого часу не змогли знайти проєкт. У першому приближенні цей набір шаблонів для правил виглядає наступним чином:

- *ЯКЩО (Команда X без відповідності*
  ТА Проєкт Y без відповідності
  ТА Пріоритет команди X = N1
  ТА Пріоритет команди Y = N2)
  ТО (У команди X є відповідність
  ТА У проєкту Y є відповідність
  ТА Команда X відповідає проєкту Y)
- *ЯКЩО (Пріоритет команди X відсутній*
  АБО Пріоритет проєкту Y відсутній)
  ТО (Неможливо знайти відповідність між командою X проєктом Y)

- *ЯКЩО (Команда X без відповідності*
  ТА Команда X скоро завершує проєкт)
  ТО (Команду X необхідно розпустити)
- *ЯКЩО (Проєкт X без відповідності)*
  ТО (Необхідно шукати нову команду)
- *Тощо.*

До цього набору правил відносяться рекомендації щодо витрат, які напряму не відносяться до процесу пошуку відповідностей між потенційними проєктами та наявними командами. До них відносяться витрати на адміністративний персонал, рекламу та активне офісне приміщення. Співвідносячи поточний розмір компанії та розмір адміністративного персоналу іноді є сенс прийняти рішення щодо скорочення або збільшення кількості адміністративного персоналу. Аналогічно, співвідносячи ризики співпраці з наявними командами з витратами на активне офісне приміщення іноді є сенс перевести команди на дистанційну роботу або знайти менше офісне приміщення. Також можна співвіднести корисність та витрати наявних рекламних компаній з метою оцінки корисності їх продовження. Неправильні рішення щодо цих моментів також можуть негативно вплинути на прибуток аутсорсингової компанії, тому було би нелогічно проігнорувати такий набір правил в рамках задачі максимізації прибутку. Зважаючи на цю інформацію можемо сформувати наступні шаблони правил:

- *ЯКЩО ([Ризик взаємодії з командою X = N]xM АБО [...]xM)*
  ТО ({Є, Немає} ризику дистанційної роботи)
- *ЯКЩО ({Є, Немає} ризику дистанційної роботи*
  ТА Витрати на офіс = N1)
  ТО ({Є, Немає} сенсу переведення на дистанційну роботу)
- *ЯКЩО (Розмір компанії = N1*
  ТА Розмір адміністративного персоналу = N2)
  ТО (Необхідно {Зменшити, Ніякий дій, Збільшити} адміністративний персонал)

- *ЯКЩО (Витрати на рекламну кампанію типу X = N1*
  ТА Користь від рекламної компанії типу X = N2)
  ТО (Необхідно {Зменшити, Ніякий дій, Збільшити} рекламну кампанію типу X)
- *Тощо.*

Змоделюємо тестову ситуацію для рішення задачі максимізації прибутку аутсорсингової ІТ-компанії. Поточна ситуація за проєктами полягає у наявності 5 проєктів у розробці, що у свою чергу означає наявність 5 команд для розробки потенційних проєктів. Розглянемо більш детально інформацію за цими командами у табл. 3.5.

Таблиця 3.5 – Наявні команди (Джерело – власна розробка)

Назва	**Ризик співпраці**	**Рівень компенсації**	**Завершення проєкту (у місяцях)**
1	2	3	4
CycleDuo	Низький	Вище середнього	12
Templater	Низький	Нижче середнього	15
Avion	Середній	Висока	3
Howl	Низький	Вище середнього	4
Converge	Низький	Низька	5

Далі розглянемо поточну ситуацію на ринку щодо наявних для вибору проєктів та їх замовників. Детальна інформація за потенційними проєктами та їх замовниками у табл. 3.6 та 3.7.

Таблиця 3.6 – Потенційні проєкти (Джерело – власна розробка)

Назва	**Замовник**	**Тип прибутку**	**Рівень прибутку**
1	2	3	4
Genesis	Mazzle	Не фіксований	Високий
Crowding	Mazzle	Не фіксований	Середній
Firantis	Global State	Фіксований	Вище середнього
Exploration	Global State	Не фіксований	Низький
Hymera	Global State	Не фіксований	Високий

Таблиця 3.7 – Потенційні замовники (Джерело – власна розробка)

Назва	Поточний замовник	Рівень відхилення оцінок	Ризик співпраці
1	2	3	4
Mazzle	Так	Інформація відсутня	Високий
Global State	Ні	Проєкти виконуються довше запланованого	Низький

На останок розглянемо додаткову інформацію щодо витрат напряму не пов'язаних з поточними проєктами, а саме витрати пов'язані з утриманням офісу, адміністративного персоналу та витрати на різноманітні рекламні кампанії. Інформацію щодо рекламних компаній можемо переглянути у табл. 3.8. Інші витрати можемо переглянути у табл. 3.9.

Таблиця 3.8 – Рекламні кампанії (Джерело – власна розробка)

Тип кампанії	Корисність	Рівень витрат
1	2	3
Соціальні мережі	Так	Низький
Конференції	Ні	Високий
Університети	Так	Середній
Зовнішня реклама	Ні	Високий

Таблиця 3.9 – Інші витрати (Джерело – власна розробка)

Розмір компанії	Розмір адміністративного персоналу	Рівень витрат на офіс
1	2	3
Середній	Високий	Середній

Базуючись на попередньо сформульованих шаблонах побудови правил, формується набір правил для нечіткої бази знань, які будуть використовуватися для рішення задачі максимізації прибутку.

На основі даних, представлених в табл. 3.5 – 3.9, формуються початкові дані, необхідні для вирішення задачі.

Перейдемо до проведення нечіткого виводу та рішення задачі максимізації прибутку у типовій аутсорсинговій ІТ-компанії з використанням

програмної оболонки. Спочатку додаємо новий профіль та заповнимо базу правилами, сформованими на основі шаблонів представлених у цьому розділі. Результат заповнення можемо бачити на рис. 3.11.

Далі сформуємо початкові дані на основі інформації з табл. 3.5 – 3.9 та почнемо процес нечіткого виводу.

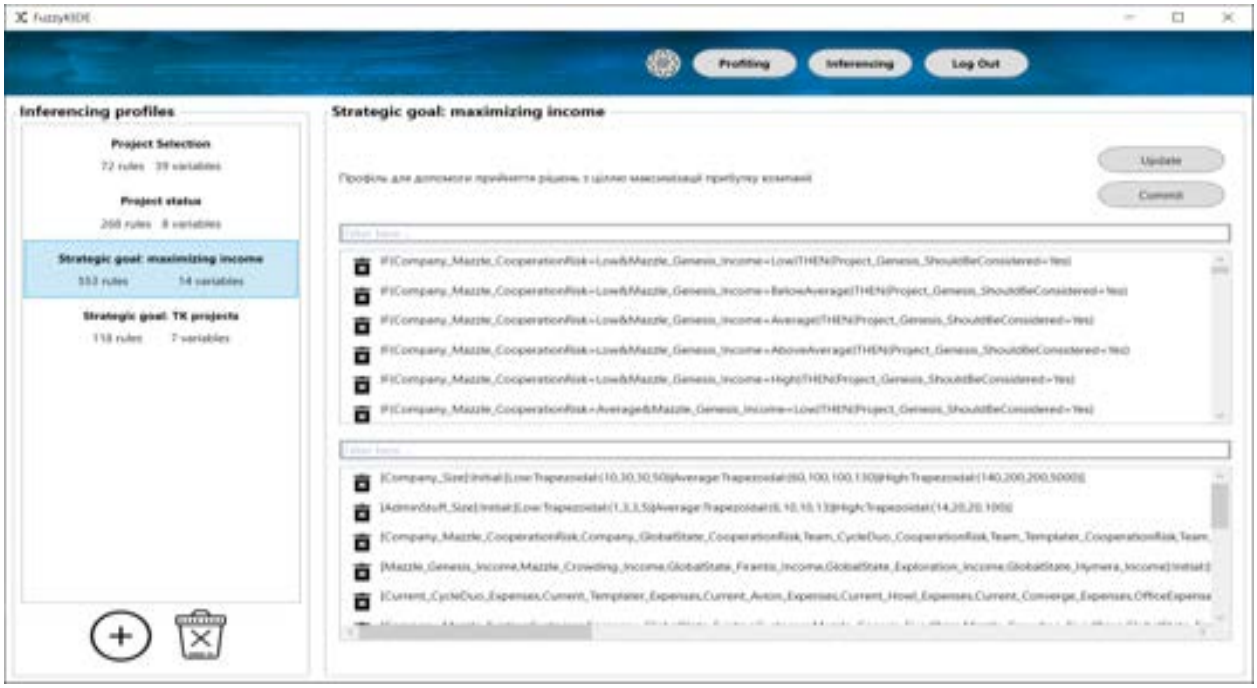


Рисунок 3.11 – Заповнена база знань для задачі максимізації прибутку (Джерело – власна розробка)

Будемо мати на увазі, що результат виводу цієї задачі є багатофакторним та включає в себе в такі результати та рекомендації:

- Пари відповідностей наявних команд та потенційних проєктів;
- Найму нових команд та розпуску існуючих;
- Відмови від потенційних проєктів;
- Регулювання розміру адміністративного персоналу;
- Врегулювання ситуації з орендою офісного приміщення;
- Регулювання вкладень в поточні рекламні кампанії.

Розглянемо кожен з цих пунктів детальніше. На рис. 3.12 – 3.15 наведено результати нечіткого виводу щодо знаходження відповідностей між проєктами та командами:

- *Hymera Converge Matched = Yes (60) was enabled with confidence factor 0.7;*
- *Exploration Converge Matched = Yes (60) was enabled with confidence factor 0.7;*
- *Firantis Avion Matched = Yes (60) was enabled with confidence factor 0.7;*
- *Genesis Howl Matched = Yes (60) was enabled with confidence factor 0.8.*

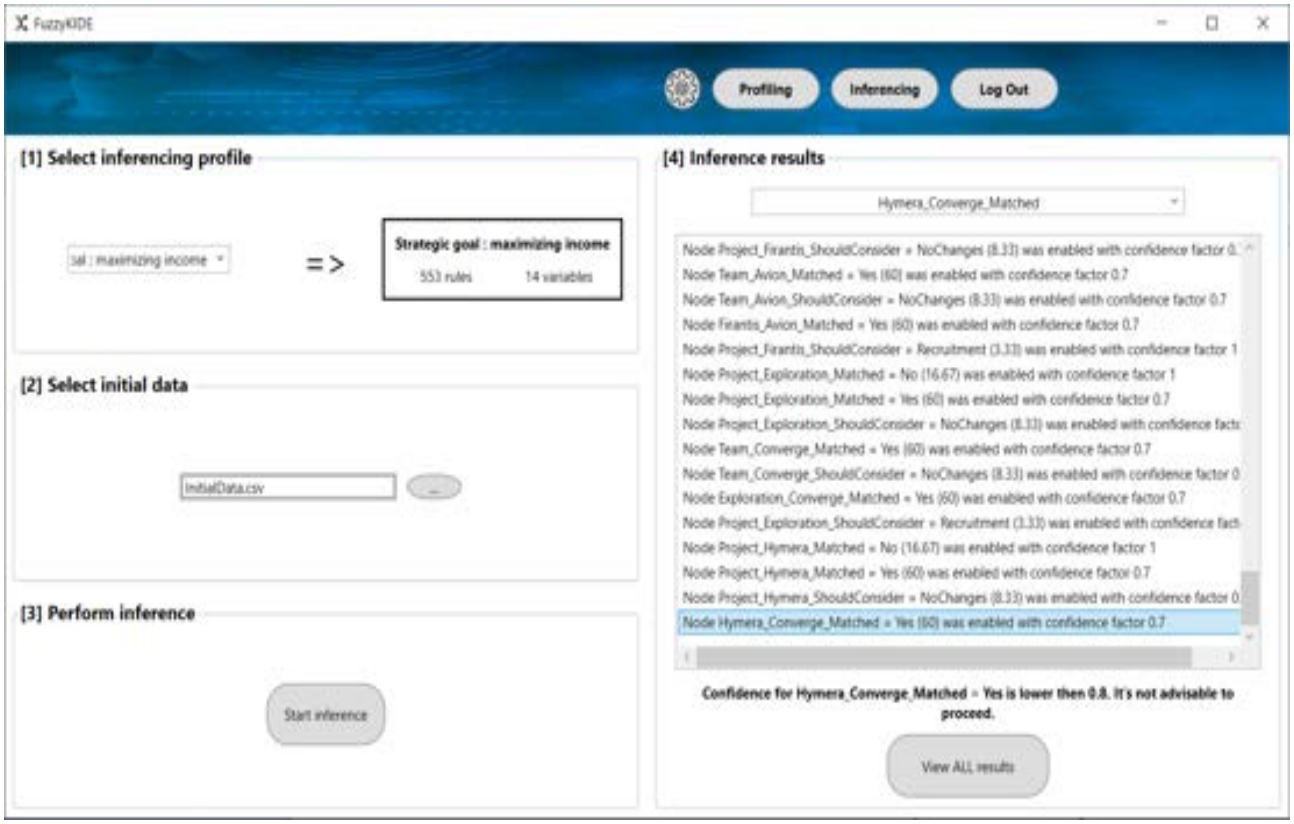


Рисунок 3.12 – Відповідність: Hymera та Converge (Джерело – власна розробка)

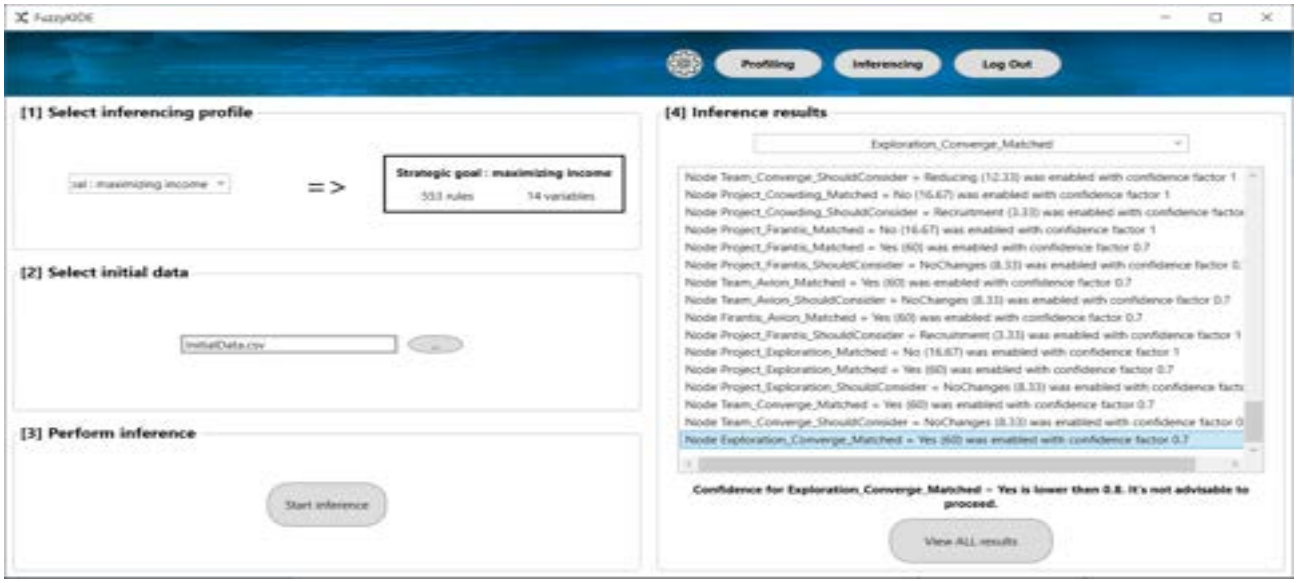


Рисунок 3.13 – Відповідність: Exploration та Converge

(Джерело – власна розробка)

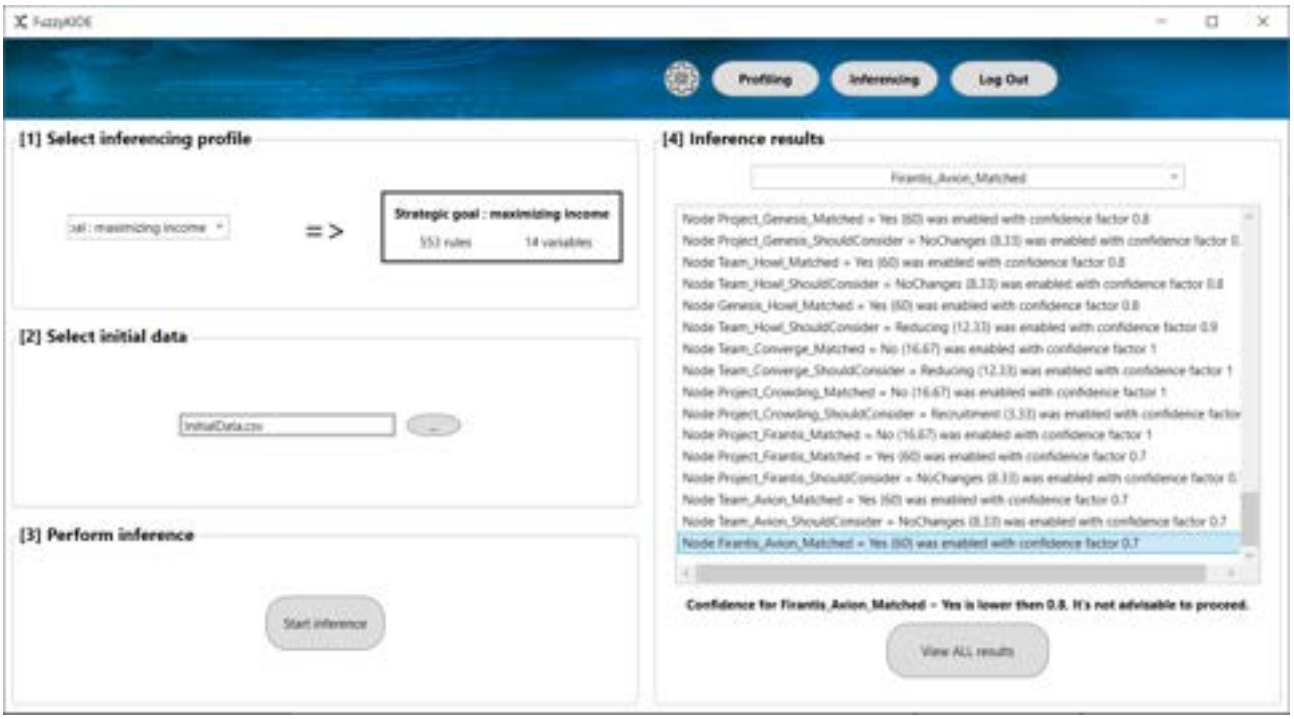


Рисунок 3.14 – Відповідність: Firantis та Avion (Джерело – власна розробка)

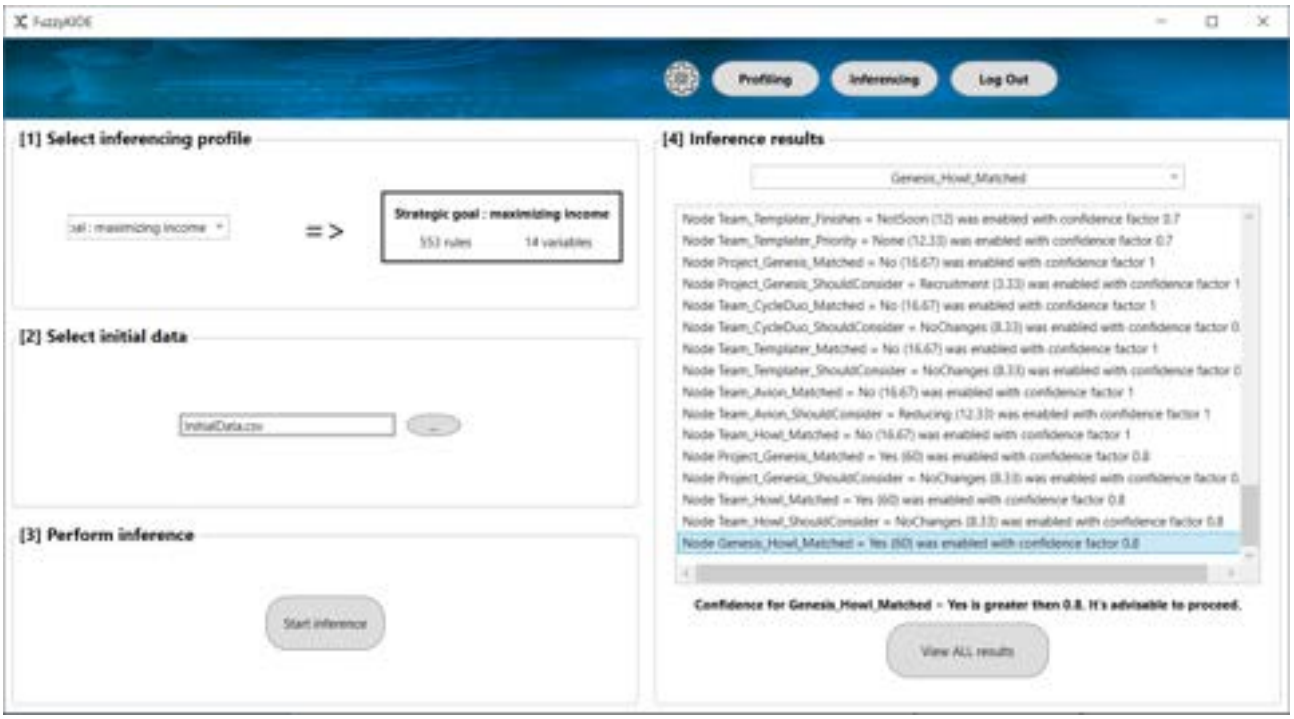


Рисунок 3.15 – Відповідність: Genesis та Howl (Джерело – власна розробка)

Вузол Hymera *Converge Matched = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що менше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система була в змозі

знайти відповідність між потенційним проєктом Hymera та наявною командою Converge, але не рекомендує дій щодо цього рішення. Якщо розглянути процес виводу детальніше, то можемо бачити, що така ситуація склалася через комбінацію невизначеності щодо низького ризику співпраці з командою Converge та високого ступеню відхилення оцінок тривалості проєктів компанії Global State, яка пропонує проєкт Hymera.

Вузол *Exploration Converge Matched = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що менше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система була в змозі знайти відповідність між потенційним проєктом Exploration та наявною командою Converge, але не рекомендує дій щодо цього рішення. Якщо розглянути процес виводу детальніше, то можемо бачити, що виникає ситуація досить схожа з попередньою. Немає впевненості щодо низького ризику співпраці з командою Converge і, так як, проєкт Exploration також пропонується компанією Global State, то немає впевненості щодо високого ступеню відхилення оцінок тривалості проєктів цієї компанії.

Вузол *Firantis Avion Matched = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що менше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система була в змозі знайти відповідність між потенційним проєктом Firantis та наявною командою Avion, але не рекомендує дій щодо цього рішення.

У процесі розгляду результатів виводу, бачимо, що впевненості у прийнятому рішенні знову немає через належність проєкту Firantis компанії Global State та невпевненості у часових оцінках тривалості, створюємих цією компанією.

Вузол *Genesis Howl Matched = Yes (60)* було активовано з коефіцієнтом достовірності 0.8, що дорівнює пороговому значенню за замовчуванням. Це свідчить про те, що експертна система була в змозі знайти відповідність між потенційним проєктом Genesis та наявною командою Howl та впевнена у цьому рішенні.

В усіх отриманих результатах нечіткому значенню *Yes* відповідає чітке значення 60. Щодо інших результатів, то проєкту Crowding не вдалося знайти успішної відповідності. Якщо подивитися на стан поточних проєктів, то можна бачити, що насправді тільки три проєкти доступні для процесу пошуку відповідностей через те, що проєкти CycleDuo та Templater не скоро закінчуються. І команді Converge вдалося знайти відповідність одразу на два проєкти: Hymera та Exploration.

З цим результатом пов'язане інше рішення експертної системи. Якщо у результатах звернути увагу на пропозиції щодо початку процесу пошуку нових команд для потенційних проєктів, то можемо бачити, що система не пропонує почати цей процес для проєктів Hymera або Exploration, адже система була в змозі знайти їм відповідність, хоча, логічно, що так як ця відповідність – одна команда, то тільки один проєкт зможе її використати. Для виправлення цієї ситуації експертові необхідно прийняти рішення та вилучити обрану відповідність з початкових даних.

Щодо проєкту Crowding такої ситуації не виникає і експертна система рекомендує почати пошук нової команди для цього проєкту. Також система не дає пропозицій про розпуск наявних команд, тому що вони або будуть зайняті на поточних проєктах впродовж тривалого часу, або їм успішно вдалося знайти відповідність у потенційно нових проєктах.

Далі розглянемо результати за іншими видами витрат. На рис. 3.16 – 3.19 можемо бачимо пропозиції щодо поточних рекламних кампаній:

- *Advertisement Social Media Should Consider = Recruitment (3.33) was enabled with confidence factor 0.9;*
- *Advertisement Conferences Should Consider = Reducing (12.33) was enabled with confidence factor 0.7;*
- *Advertisement Universities Should Consider = NoChanges (8.33) was enabled with confidence factor 0.8;*
- *Advertisement Outdoor Should Consider = Reducing (12.33) was enabled with confidence factor 1.*

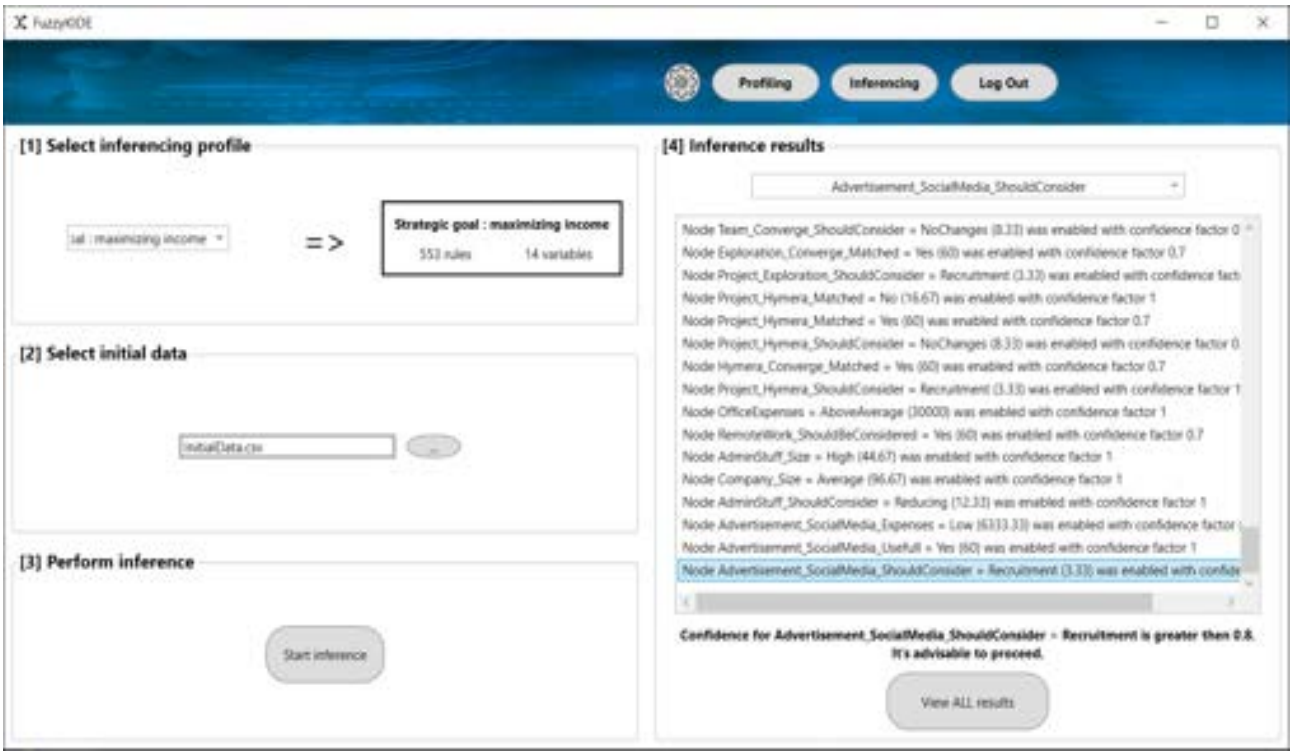


Рисунок 3.16 – Рекламна кампанія: соціальні мережі

(Джерело – власна розробка)

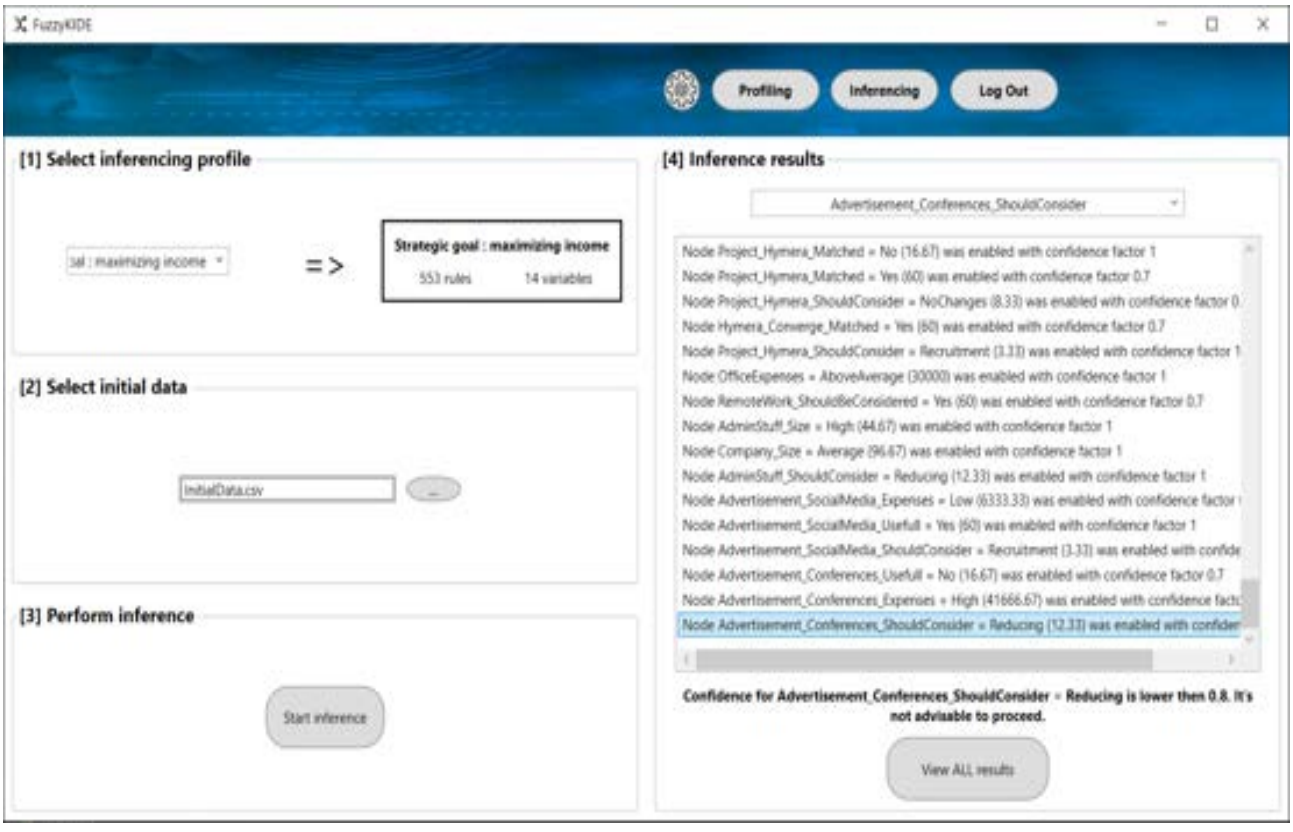


Рисунок 3.17 – Рекламна кампанія: конференції

(Джерело – власна розробка)

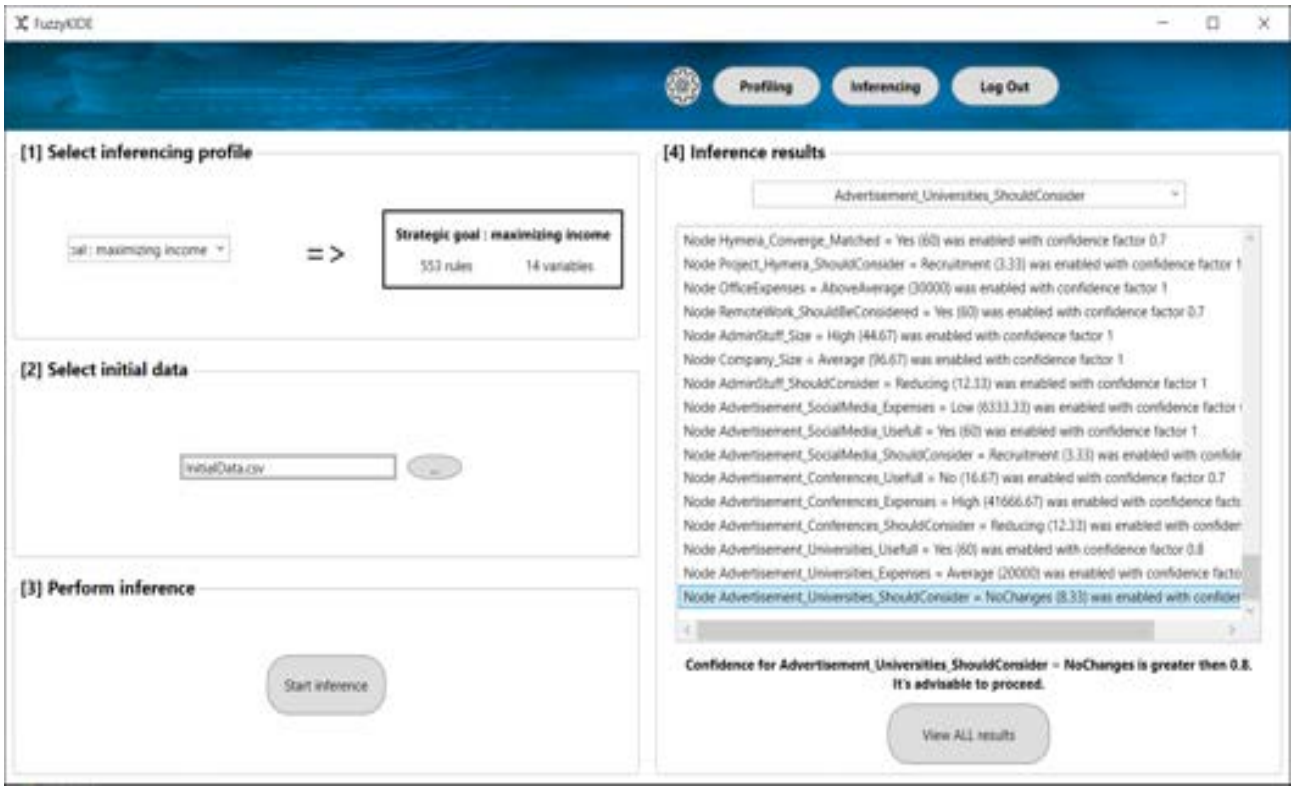


Рисунок 3.18 – Рекламна кампанія: університети (Джерело –власна розробка)

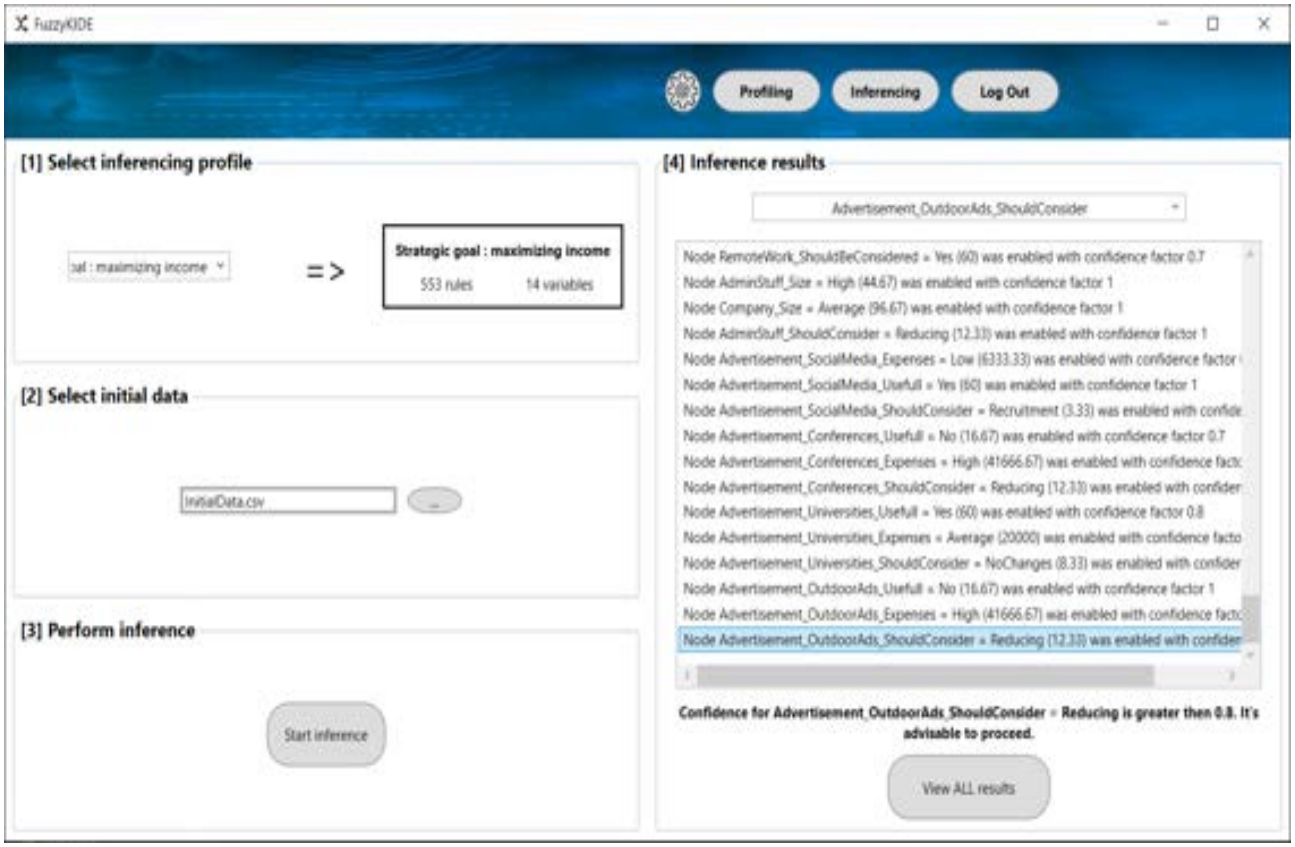


Рисунок 3.19 – Рекламна кампанія: зовнішня реклама

(Джерело – власна розробка)

Вузол *Advertisement Social Media Should Consider = Recruitment (3.33)* було активовано з коефіцієнтом достовірності 0.9, що більше порогового значенню за замовчуванням (0.8). Це свідчить про те, що експертна система дає рекомендацію збільшити витрати на рекламну кампанію в соціальних мережах та впевнена у цьому рішенні. Таке рішення було прийнято через комбінацію низьких витрат на такий тип рекламної кампанії та високий рівень її корисності.

Вузол *Advertisement Conferences Should Consider = Reducing (12.33)* було активовано з коефіцієнтом достовірності 0.7, що менше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система дає рекомендацію зменшити витрати на рекламну кампанію посередством конференцій, але не впевнена у цьому рішенні. Таке рішення було прийнято через високий рівень витрат на конференції та низький рівень її корисності. Щодо невпевненості у прийнятому рішенні, то можемо бачити, що така ситуація склалася через невпевненість щодо відсутньої корисності цього типу реклами.

Вузол *Advertisement Universities Should Consider = No Changes (8.33)* було активовано з коефіцієнтом достовірності 0.8, що дорівнює пороговому значенню за замовчуванням. Це свідчить про те, що експертна система дає рекомендацію залишити рекламну кампанію посередством університетських ярмарок зайнятості та інших пов'язаних активностей без змін та впевнена у цьому рішенні. Таке рішення було прийнято через комбінацію середнього рівня витрат на такий тип рекламної кампанії та високий рівень її корисності.

Вузол *Advertisement Outdoor Should Consider = Reducing (12.33)* було активовано з коефіцієнтом достовірності 1, що більше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система дає рекомендацію зменшити витрати на зовнішню рекламу та впевнена у цьому рішенні. Таке рішення було прийнято через комбінацію високих витрат на зовнішню рекламу та низький рівень її корисності. В усіх результатах можемо

бачити, що нечітким значенням *Recruitment, No Changes* та *Reducing* відповідають чіткі значення 3.33, 8.33 та 12.33 відповідно.

Наостанок розглянемо 2 результати, пов'язані з витратами на офісне приміщення та адміністративний персонал підтримки.

На рис. 3.20 та 3.21 можемо бачити результати виводу щодо цих витрат:

- *Admin Stuff Should Consider = Reducing (12.33) was enabled with confidence factor 1;*
- *Remote Work Should Be Considered = Yes (60) was enabled with confidence factor 0.7.*

Вузол *Admin Stuff Should Consider = Reducing (12.33)* було активовано з коефіцієнтом достовірності 1, що вище порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система дає рекомендацію зменшити розмір адміністративного персоналу та впевнена у цьому рішенні.

Таке рішення було прийнято через невідповідність розмірів адміністративного персоналу та загального розміру компанії, а саме надлишку адміністративного персоналу. Нечіткому значенню *Reducing* відповідає чітке значення 12.33.

Вузол *Remote Work Should Be Considered = Yes (60)* було активовано з коефіцієнтом достовірності 0.7, що менше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система дає рекомендацію знайти менше офісне приміщення з дешевшою орендою або взагалі відмовитися від офісу на користь дистанційної роботи, але не впевнена у цьому рішенні.

Експертна система проаналізувала ризики взаємодії з існуючими командами та прийняла рішення, що загалом по компанії цей ризик низький. В комбінації з високою орендною платою і було надано рекомендацію щодо перегляду поточної ситуації з офісним приміщенням. Нечіткому значенню *Yes* відповідає чітке значення 60.

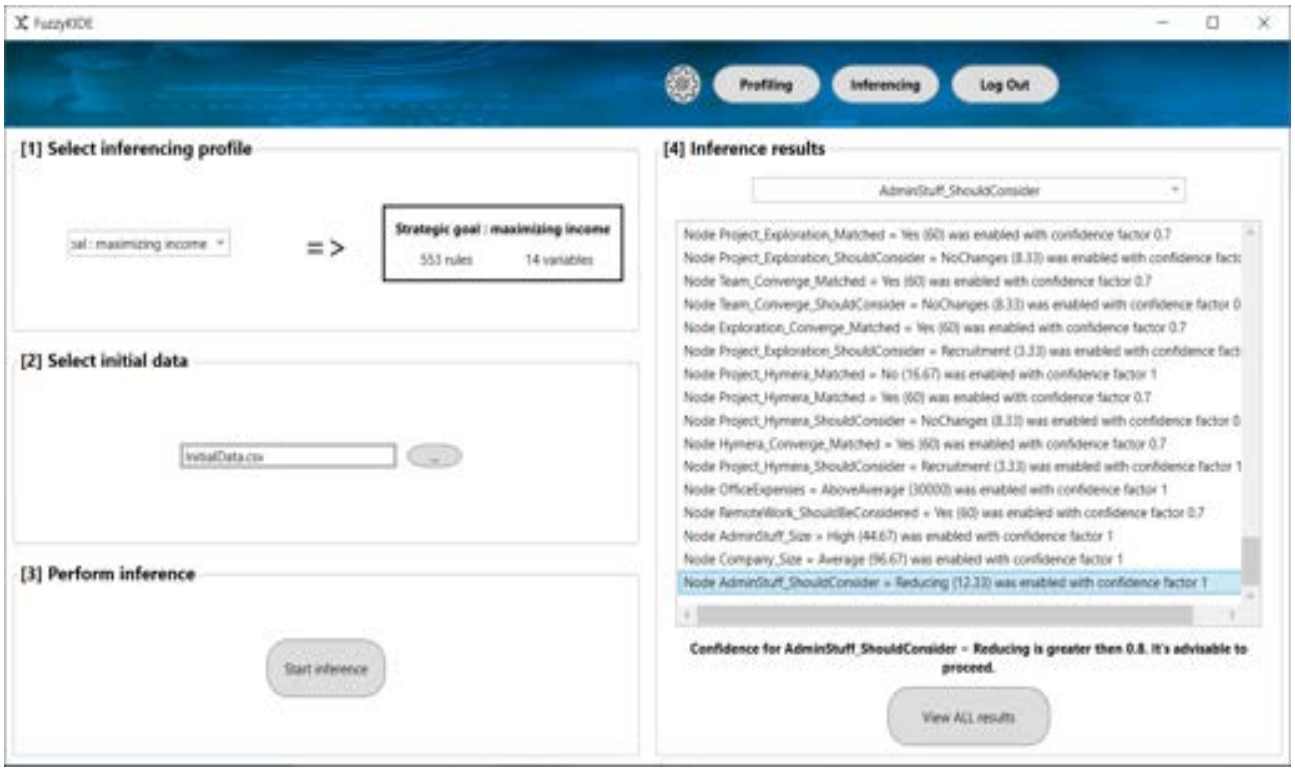


Рисунок 3.20 – Інші витрати: адміністративний персонал

(Джерело –власна розробка)

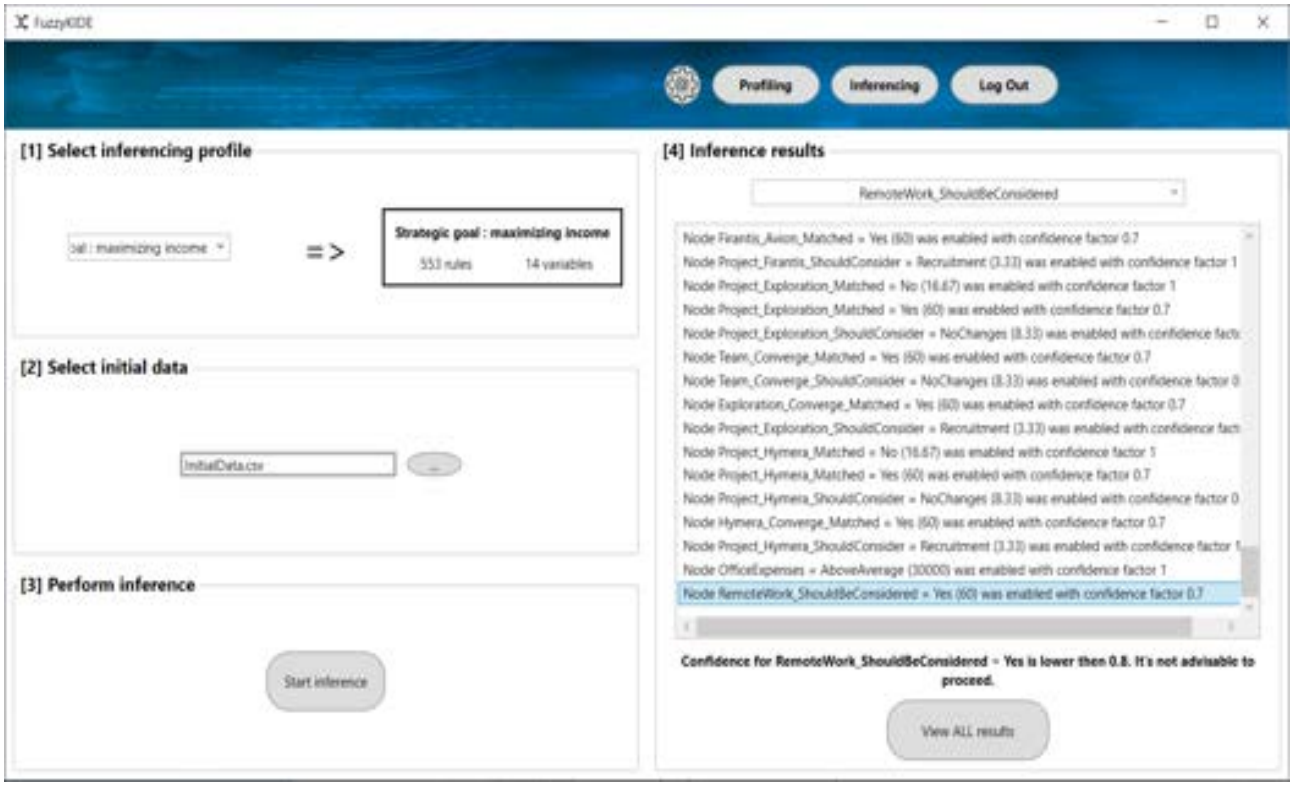


Рисунок 3.21 – Інші витрати: офісне приміщення

(Джерело –власна розробка)

3.2 Розробка економічних рішень з формування портфоліо проєктів

Розглянемо ще один приклад використання програмної оболонки, а саме розглянемо задачу пошуку проєкту аутсорсингової ІТ-компанії. Ця задача, як правило, є першою в циклі розробки програмного забезпечення в ІТ-компанії [3].

Для підтримки та стимулювання фінансових потоків перед аутсорсинговою ІТ-компанією майже завжди постає задача пошуку подальших проєктів для виконання. Зважаючи, що рішення стосовно прийняття проєкту може прийматися не однією людиною, а групою, може бути досить складно не упустити частину наявної інформації в процесі комунікації щодо прийняття цього рішення. Для спрощення процесу прийняття рішення щодо вибору проєкту доцільно мати експертну систему з наявними даними, яка зможе дати деякі рекомендації усім учасникам цього процесу стосовно проєктів, які розглядаються.

Для формування бази знань для цієї задачі спочатку необхідно розглянути найважливіші складові, які постають перед менеджментом при прийнятті таких рішень:

- Замовник, який пропонує проєкт.
- Співробітники ІТ-компанії, яка є об’єктом дослідження.
- Потреби запропонованого проєкту.

Перший набір правил до бази знань формується на основі наявних даних щодо замовників, які пропонують проєкти. Найважливішими аспектами цього питання є різноманітні фінансові та нефінансові ризики роботи з замовниками, власний або запозичений попередній досвід роботи з замовниками. Якщо детальніше, то можна розглянути наступні питання щодо вибору замовника:

- Оцінка ризиків роботи з замовником:
  - внутрішні історичні дані;
  - зовнішні історичні дані;

- відгуки інших IT-компаній;
- Загальнодоступні рейтинги;

- Кореляційні дані щодо роботи з подібними замовниками:
- Врахування культурних особливостей замовника.

Для побудови правил оцінки ризиків роботи з замовниками можна використати набір правил KYC (Know your customer) [65] та адаптувати його до потреб наявної задачі. Спершу, використовуючи KYC, можемо сформувати правила для оцінки фінансових ризиків, пов'язаних з країною розташування головного офісу:

- *ЯКЩО (Країну перебування Замовника X було змінено)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Країна перебування Замовника X = N1)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Країна перебування Замовника X = N1)*
  ТО (Співпраця із Замовником X неможлива)
- *ЯКЩО (Країну головного офісу Замовника X було змінено)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Країна головного офісу Замовника X = N1)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Країна головного офісу Замовника X = N1)*
  ТО (Співпраця із Замовником X неможлива)
- *Тощо.*

Далі можемо сформувати правила для оцінки фінансових ризиків, пов'язаних з банківськими операціями замовника:

- *ЯКЩО (Банківський рахунок Замовника X було змінено)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Банк Замовника X було змінено в межах країни)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Банк Замовника X було змінено за межами країни)*
  ТО (Рівень ризику співпраці із Замовником X = N2)

- *ЯКЩО (Банк Замовника X = N1)*

  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Банк Замовника X = N1)*

  ТО (Співпраця із Замовником X неможлива)
- *Тощо.*

Далі можемо сформувати правила для оцінки ризиків індустрії, джерел прибутку замовника, тощо:

- *ЯКЩО (Індустрія Замовника X = N1)*

  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Індустрія Замовника X = N1)*

  ТО (Співпраця із Замовником X неможлива)
- *ЯКЩО (Замовник X в процесі зміни керівництва)*

  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Замовник X в процесі зміна керівництва)*

  ТО (Співпраця із Замовником X неможлива)
- *ЯКЩО (Країну збуту Замовника X = N1)*

  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Країну збуту Замовника X = N1)*

  ТО (Співпраця із Замовником X неможлива)
- *ЯКЩО (Основний клієнт Замовника X = N1)*

  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Основний клієнт Замовника X = N1)*

  ТО (Співпраця із Замовником X неможлива)
- *ЯКЩО (Джерело збагачення Замовника X = N1)*

  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Джерело збагачення Замовника X = N1)*

  ТО (Співпраця із Замовником X неможлива)
- *ЯКЩО (Інтенсивність бізнес процесів Замовника X = N1 ТА фінансові обсяги Замовника X = N2)*

  ТО (Рівень ризику співпраці із Замовником X = N3)

- *Тощо.*

Також наведемо приклади декількох правил, не пов'язаних з KYC:

- *ЯКЩО (З Замовником X вже співпрацювали*
  ТА Тривалість співпраці з Замовником X = N1)
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *ЯКЩО (Замовника X перевірено іншою компанією)*
  ТО (Рівень ризику співпраці із Замовником X = N2)
- *Тощо.*

Таким чином маємо приблизно сформований скелет для набору правил бази знань щодо питань пошуку замовника.

Другий набір правил до бази знань формується на основі наявної інформації щодо співробітників компанії. Проаналізувавши джерела, які спеціалізуються на оцінці співробітників [25, 82], було виділено найважливіші складові цього питання, які розглядають менеджери:

- Набір технічних навичок співробітника;
- Рівень кваліфікації співробітника;
- Історичні дані щодо участі співробітника у попередніх проєктах;
- Психологічні та культурні особливості співробітника;
- Термін знаходження співробітника у компанії.

Враховуючи цю інформацію, наведемо декілька загальних прикладів цих правил. Почнемо з технічних навичок та кваліфікації:

- *ЯКЩО (Співробітник X = Front-end розробник)*
  ТО (У наявності є Front-end розробник)
- *ЯКЩО (Співробітник X = Back-end розробник)*
  ТО (У наявності є Back-end розробник)
- *ЯКЩО (Співробітник X = Проєктний менеджер)*
  ТО (У наявності є Проєктний менеджер)
- *ЯКЩО (Рівень Співробітника X = N1)*
  ТО (У наявності є співробітник рівня N1)
- *Тощо.*

Далі перейдемо до психологічних та культурних особливостей співробітника:

- *ЯКЩО (Комунікаційні навички Співробітника X = N1)*

  ТО (Рівень ризику залучення Співробітника X до проєкту Y = N2)
- *ЯКЩО (На Співробітника X є скарги)*

  ТО (Рівень ризику залучення Співробітника X до проєкту Y = N2)
- *Тощо.*

Далі перейдемо до наборів правил стосовно термінів співпраці співробітника:

- *ЯКЩО (Роки співпраці Співробітника X = N1)*

  ТО (Рівень ризику залучення Співробітника X до проєкту Y = N2)
- *ЯКЩО (Співробітник X вільний*

  АБО Поточний проєкт Співробітника X закінчується)

  ТО (У наявності є Співробітник X)
- *Тощо.*

Таким чином маємо приблизно сформований скелет для набору правил бази знань щодо питань пошуку співробітників.

Попередньо сформовані комбінації правил дозволяють дізнатися чи доцільно працювати з замовниками, описаними в базі знань та які співробітники є у наявності на даний момент. Але найважливішим є третій набір правил до бази знань, який формується на основі потреб проєкту. Потреби проєкту можливо поділити на наступні групи:

- Аналіз технологій, необхідних для виконання та розробки проєкту;
- Аналіз наявного персоналу на предмет відповідності кваліфікацій та поточної наявності.

Враховуючи цю інформацію, наведемо декілька загальних прикладів цих правил:

- *ЯКЩО (Проєкту необхідний Співробітник з рівнем N1*

  ТА Співробітник з рівнем N1 є у наявності)

  ТО (Потреба у Співробітнику з рівнем N1 задоволена)

- *ЯКЩО (Проєкту необхідний Back-end розробник*
  ТА Back-end розробник є у наявності)
  ТО (Потреба у Back-end розробнику задоволена)
- *ЯКЩО (Потреба у Back-end розробнику задоволена*
  ТА Потреба у Співробітнику з рівнем N1 задоволена)
  ТО (Потреба у співробітниках задоволена)
- *ЯКЩО (Потреба у співробітниках задоволена*
  ТА співпраця з Замовником X можлива)
  ТО (Можливо почати роботу над проєктом Y)
- *Тощо.*

Таким чином маємо приблизно сформований скелет для набору правил бази знань щодо питань відповідності ризиків роботи з замовником, відповідності та наявності співробітників до потреб проєкту. Це дає можливість розглянути більш детальний приклад.

Змоделюємо тестову ситуацію для демонстрації рішення задачі пошуку проєкту в рамках аутсорсингової ІТ-компанії. Припустимо, що для розгляду доступна невелика аутсорсингова ІТ-компанія. Зі сторони компанії у загалом маємо 8 співробітників різного рівня розвитку та технічних і нетехнічних навичок. Розглянемо інформацію про наявних співробітників більш детальніше у табл. 3.10 та 3.11.

Таблиця 3.10 – Співробітники – навички (Джерело – власна розробка)

Ім'я	**Технічні навички та рівень кваліфікації**	**Комунікаційні навички**
1	2	3
Alex	Senior Full-stack developer	Добре розвинуті
Bob	Middle Front-end developer	Добре розвинуті
Marco	Junior Back-Stack developer	Погано розвинуті
Polo	Junior Front-Stack developer	Добре розвинуті
Miguel	Junior Back-Stack developer	Добре розвинуті
Susan	Middle Automation QA	Погано розвинуті
Fernando	Junior Manual QA	Добре розвинуті
Rick	Middle project manager	Дуже добре розвинуті

Таблиця 3.11 – Співробітники – додаткова інформація (Джерело – власна розробка)

Ім'я	Культурні особливості	Роки співпраці	Доступність
1	2	3	4
Alex	Скарг немає	2	Так
Bob	Антисеміт, скарг немає	1	Так
Marco	Є скарги	1	Так
Polo	Скарг немає	0.5	Так
Miguel	Скарг немає	0.5	Так
Susan	Скарг немає	1	Так
Fernando	Расист, є скарги	0.5	Ні
Rick	Скарг немає	2	Так

Зі сторони замовників маємо 2 замовника. Global State є перевіреним поточним клієнтом компанії без підозрілих фінансових та банківських операцій. З іншої сторони Mazzle є новим клієнтом компанії з підозрілим фінансовим потоком. Опишемо інформацію про замовників більш детальніше у табл. 3.12 – 3.14.

Попередньо описані замовники мають у загалом 5 проєктів на вибір. Опишемо детальніше потреби проєктів у табл. 3.15.

Таблиця 3.12 – Замовники – основна інформація (Джерело – власна розробка)

Компанія	Банк	Країна	Індустрія	Основний клієнт
1	2	3	4	5
Global State	Не доступно	Нідерланди	Телекомунікація	Не доступно
Mazzle	Deutsche bank	Ізраїль	Спостереження	Microsoft

Таблиця 3.13 – Замовники – історична інформація (Джерело – власна розробка)

Компанія	Новий замовник	Онлайн рейтинг
1	2	3
Global State	Ні	Високий
Mazzle	Так	Середній

Таблиця 3.14 – Замовники – додаткова інформація (Джерело –власна розробка)

Компанія	Поточні зміни	Грошовий потік
1	2	3
Global State	Зміна країни головного офісу : Ні; Зміна банку в межах країни : Не доступно; Зміна банку за межами країни: Не доступно	Бізнес з інтенсивним грошовим потоком : Так; Прибуток відповідає інтенсивності : Так
Mazzle	Зміна країни головного офісу : Ні; Зміна банку в межах країни : Ні; Зміна банку за межами країни: Ні	Бізнес з інтенсивним грошовим потоком : Так; Прибуток відповідає інтенсивності : Ні

Таблиця 3.15 – Проєкти (Джерело – власна розробка)

Компанія	Проєкт	Потреби
1	2	3
Global State	Firantis	Senior developer
	Fabolous	Senior developer; Senior QA
	Exploration	Middle developer; Middle project manager; Front-end developer
Mazzle	Genesis	Senior developer; Senior project manager; Automation QA
	Crowding	Junior developer; Junior QA

Базуючись на інформації щодо співробітників, замовників та проєктів, представленій у цьому розділі та на прикладах формування правил формується набір нечітких правил до бази знань, які будуть використовуватись для рішення задачі пошуку проєкту.

Також на основі цієї інформації буде сформовано початкові дані, необхідні для вирішення цієї задачі.

Продемонструємо рішення задачі пошуку проєкту у типовій аутсорсинговій ІТ-компанії, використовуючи програмну оболонку. Для рішення цієї задачі спершу додаємо новий профіль та заповнимо базу знань на основі продемонстрованих раніше шаблонів правил – рис. 3.22. Після цього вкажемо початкові дані, сформовані на основі інформації з табл. 3.10 – 3.15.

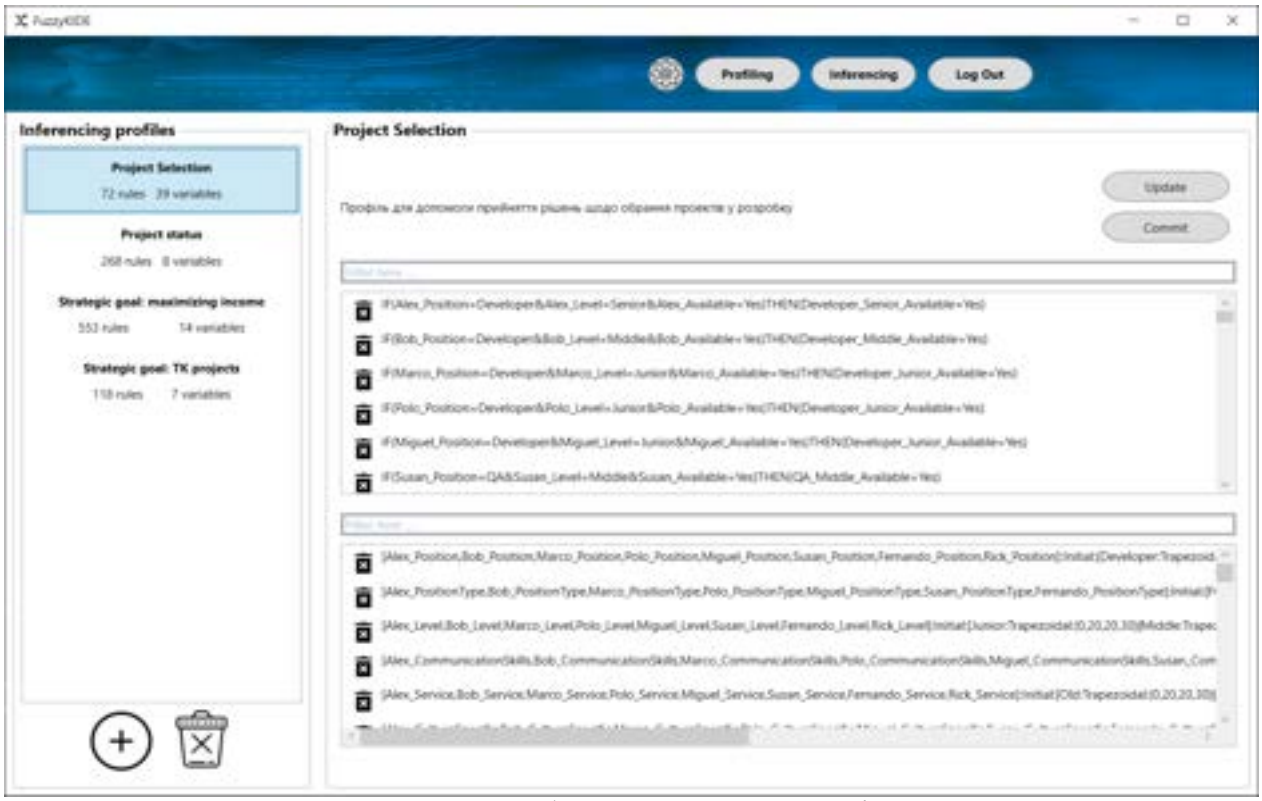


Рисунок 3.22 – Заповнена база знань для задачі пошуку проєкту (Джерело – власна розробка)

На рис. 3.23 наведено результат нечіткого виводу стосовно задачі пошуку проєкту: *Mazzle Crowding OK = Yes (60) was enabled with confidence factor 0.5.*

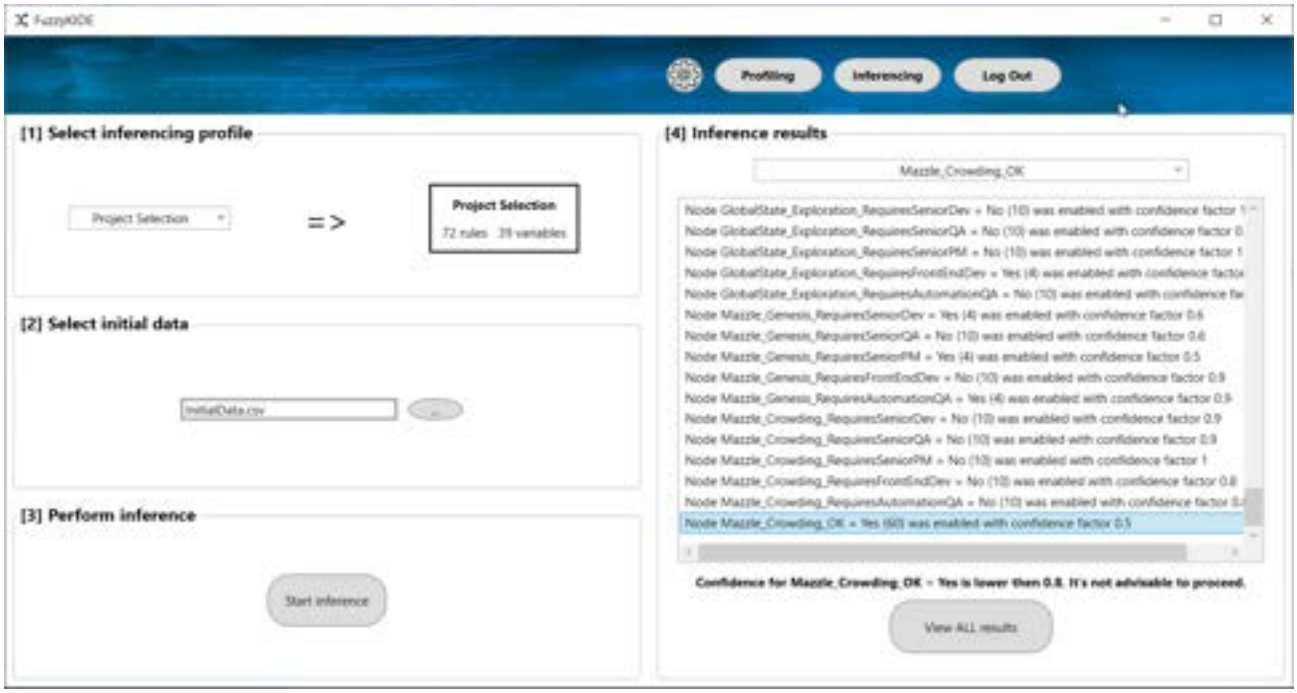


Рисунок 3.23 – Рішення задачі пошуку проєкту (Джерело – власна розробка)

Як бачимо, програмна оболонка дає рекомендацію щодо уникнення старту проєкту Crowding замовника Mazzle. Вивід було проведено таким чином, що вузол *Mazzle Crowding OK = Yes (60)* було активовано с коефіцієнтом достовірності 0.5, що менше порогового значення за замовчуванням (0.8). Такий результат свідчить про достатню кількість інформації щодо проєкту Crowding, але досить низький ступінь довіри до цієї інформації у початкових даних. Нечіткому значенню *Yes* відповідає чітке значення 60. Також аналізуючи інші результати виводу, можна побачити, що вузли, відповідаючі за рекомендації щодо старту інших чотирьох проєктів не були активовані під час процесу виводу, що свідчить про нестачу правил у сформованій базі знань або початкових даних, щоб прийти до цих результатів.

3.3 Методи моніторингу поточного статусу проєктів/портфоліо проєктів

Розглянемо ще один детальний приклад використання програмної оболонки, а саме задачу переоцінки поточного статусу проєкту аутсорсингової IT-компанії [100]. Відповідно до розділу 2.4, ця задача є основною в циклі розробки програмного забезпечення, адже є ітеративною та, як правило, найбільш тривалою. Перед аутсорсинговою компанією завжди лежить задача моніторингу поточного статусу проєкту з ціллю своєчасного реагування на несприятливі зміни у процесі виконання проєкту. Існує досить багато причин, які можуть призвести до небажаних результатів та потребують оперативного реагування задля уникнення фінансових скрут, які стосуються розглядаємого проєкту. Найбільш вірогідні причини:

- Перевитрати на членів команди;
- Командне відставання по товарному запасу продукту;
- Неправильно розставлені пріоритети в товарному запасі продукту.

З точки зору аутсорсингової IT-компанії, для врегулювання несприятливих змін одними з вірогідних варіантів рішення є:

- Перерозподілення персоналу з ціллю компенсації економічних показників, які призводять до несприятливих змін.
- Почати переговори з замовником щодо розширення бюджету з ціллю розширення персоналу, кількості команд або компенсації економічних показників.
- Почати переговори з замовником щодо розширення строків виконання проєкту.

Прийняте рішення в цій задачі, як правило, є багатофакторним, адже стосується та впливає на портфоліо проєктів, яке складається з декількох проєктів у рамках одного продукту. Також можлива ситуація впливу на проєкти за межами розглядаємого портфоліо проєктів.

Перший набір правил до бази знань формується на основі наявних даних щодо командного бюджету. Досить частою є ситуація перевитрат на команду. В таких випадках постає задача виявлення можливості перетасування членів команд на основі середнього професійного рівня, бюджету та статусу проєктів інших команд з ціллю вирівнювання командних бюджетів. У випадках, коли це неможливо, скоріше за все, є необхідність вирішення питання збільшення бюджету зі сторони замовника.

Сформулюємо ці правила:

- ЯКЩО (Проєкту X не вистачає бюджету на підтримку команди
  ТА Проєкт {Y, Z, etc.} має можливість перетасувати команду)
  ТО (Необхідно розпочати переговори щодо перетасування команди
  Проєкту X)
- ЯКЩО (Проєкту X не вистачає бюджету на підтримку команди
  ТА Проєкти {Y, Z, etc.} не мають можливість перетасувати команду)
  ТО (Необхідно розпочати переговори щодо розширення бюджету
  Проєкту X)
- Тощо.

Цей набір правил є досить скромним у порівнянні з наборами правил, які розглянемо далі. Але треба розуміти, що цей набір правил сильно залежить від правил розрахунку можливості перетасування команд, які й розглянемо далі.

Другий набір правил до бази знань формується на основі наявних даних щодо різноманітних причин відставання за строками виконання та об'ємами виконання. В випадках відставання знову постає задача виявлення можливості перетасування команд на основі середнього професійного рівня команд, але цього разу з ціллю прискорити виконання проєктів у разі збільшення об'єму роботи або скоротити відставання у разі його наявності. У разі неможливості знайти варіант перетасування команд є необхідність вирішення питання розширення запланованих строків або командного розширення зі сторони замовника.

Сформулюємо ці правила:

- *ЯКЩО (У Проєкту Х немає надлишку роботи або він нижче деякого рівня*
  ТА у Проєкту Х немає товарного запасу або він нижче деякого рівня)
  ТО (Проєкт Х виконується достроково)
- *ЯКЩО (У Проєкту Х надлишок роботи вище деякого рівня*
  ТА у Проєкту Х немає товарного запасу або він нижче деякого рівня)
  ТО (Проєкт Х виконується вчасно)
- *ЯКЩО (У Проєкту Х надлишок роботи вище деякого рівня*
  ТА у Проєкту Х товарний запас вище деякого рівня)
  ТО (Проєкт Х виконується с затримкою)
- *ЯКЩО (Проєкт Х виконується с затримкою)*
  ТО (У Проєкту Х немає можливості віддати співробітників)
- *ЯКЩО (Проєкт Х виконується достроково)*
  ТО (У Проєкту Х є можливість віддати співробітників)
- *ЯКЩО (Середній рівень команди Проєкту Х нижче деякого рівня)*
  ТО (У Проєкту Х немає можливості віддати співробітників)

- *ЯКЩО (Середній рівень команди Проєкту X більше деякого рівня)*
  ТО (У Проєкту X є можливість віддати співробітників)
- *ЯКЩО (У Проєкту X є необхідність у перетасуванні команд*
  ТА Проєкт {Y, Z, etc.} має можливість віддати співробітників)
  ТО (Необхідно розпочати переговори щодо перетасування команди Проєкту X)
- *ЯКЩО (У Проєкту X є необхідність у перетасуванні команд*
  ТА Проєкти {Y, Z, etc.} не мають можливості віддати співробітників)
  ТО (Необхідно розпочати переговори щодо розширення команди Проєкту X)
- *Тощо.*

Цей набір правил є дуже суттєвим, адже рішення щодо розширення та перетасування команд у випадку несприятливих змін є найбільш дієвими і швидко виконуємим, адже дуже рідко потребують великої кількості бюрократичних рішень та переговорів з замовником та сприяють швидкому вирівнюванню показників, за якими розраховується поточний статус проєкту.

Останній набір правил до бази знань базується на основі наявних даних щодо пріоритетності розробки окремих частин проєкту, розподілених між командами. Вірогідною є ситуація, коли команди, розробляючі найбільш критичні частини продукту, відстають від строків. В таких випадках знову постає задача виявлення можливості перетасування команд на основі пріоритетності окремих проєктів у рамках цього продукту з ціллю вирівняти строки відставання. Розглянемо ці правила:

- *ЯКЩО (Проєкт X виконується вчасно*
  ТА пріоритет Проєкту X низький)
  ТО (У Проєкту X є можливість віддати співробітників)
- *ЯКЩО (Проєкт X виконується вчасно*
  ТА пріоритет Проєкту X не низький)
  ТО (У Проєкту X немає можливості віддати співробітників)

- *ЯКЩО (Проєкт Х виконується с затримкою*
  ТА пріоритет Проєкту Х низький)
  ТО (Немає необхідності перетасування команд заради Проєкту Х
  ТА Немає необхідності розширення команди Проєкту Х
  ТА Немає необхідності розширення строків Проєкту Х)
- *ЯКЩО (Проєкт Х виконується с затримкою*
  ТА пріоритет Проєкту Х не низький)
  ТО (Є необхідність перетасування команд заради Проєкту Х
  АБО Є необхідність розширення команди Проєкту Х
  АБО Є необхідність розширення строків Проєкту Х)
- *Тощо.*

Цей набір правил органічно доповнює другий набір правил, адже також сильно впливає на рішення щодо розширення та перетасування команд у випадку несприятливих змін.

Змоделюємо тестову ситуацію для демонстрації рішення задачі моніторингу статусу проєкту в рамках аутсорсингової ІТ-компанії. Зі сторони компанії виконавця маємо 4 команди, працюючі над 4 основними частинами, розробляємого продукту. Розглянемо наявну інформацію про ці команди та їх проєкти у табл. 3.16.

Таблиця 3.16 – Проєкти у портфоліо – основна інформація (Джерело – власна розробка)

Назва	**Пріоритет**	**Опис**
1	2	3
UI|UX	Високий	Проєкт по розробці дизайну інтерфейсу продукту
Portal	Високий	Проєкт по розробці інтерфейсу продукту
Migration	Низький	Проєкти по розробці сервісів для міграції поточних клієнтів на нову систему
Services	Середній	Проєкт по розробці основної логіки роботи продукту

Далі розглянемо більш детальну інформацію щодо проєктів, а саме інформацію яка може відрізнятися в залежності від часу виконання проєкту,

наприклад щодо наявних відхилень, а саме за командними бюджетами, об’ємом виконуємої роботи та командним відставанням. Для порівняння розглянемо цю інформацію у три відмінні періоди часу розробки продукту: спринт 1 – початок розробки продукту, спринт 5 – середина розробки продукту та спринт 9 – один із фінальних спринтів розробки продукту.

Почнемо з моменту старту проєкту, а саме зі спринта 1. Продемонструємо початкову проєкту інформацію у табл. 3.17.

Таблиця 3.17 – Проєкти у портфоліо – спринт 1 (Джерело – власна розробка)

Назва	**Бюджет**	**Надлишок роботи**	**Товарний запас**	**Рівень команди**
1	2	3	4	5
UI|UX	Достатній	Немає	Високий	Середній
Portal	Достатній	Немає	Високий	Високий
Migration	Достатній	Немає	Високий	Низький
Services	Достатній	Немає	Високий	Середній

Припустимо, що в процесі виконання проєктів в деякий момент часу було прийнято рішення щодо перетасування команд. Відобразимо зміни у табл. 3.18.

Таблиця 3.18 – Проєкти у портфоліо – спринт 5 (Джерело – власна розробка)

Назва	**Бюджет**	**Надлишок роботи**	**Товарний запас**	**Рівень команди**
1	2	3	4	5
UI|UX	Достатній	Немає	Середній	Середній
Portal	Достатній	Наявний	Високий	Середній
Migration	Недостатній	Немає	Середній	Середній
Services	Достатній	Наявний	Низький	Середній

Перейдемо трохи ближче до кінця виконання портфоліо проєктів. Припустимо що знову прийнято рішення щодо розширення команд. Відобразимо зміни у табл. 3.19.

Базуючись на попередньо сформульованій у цьому розділі інформації та на попередньо продемонстрованих прикладах побудови правил формується

набір нечітких правил бази знань, які будуть використовуватись для рішення задачі оцінки статусу проєктів.

Таблиця 3.19 – Проєкти у портфоліо – спринт 9 (Джерело – власна розробка)

Назва	Бюджет	Надлишок роботи	Товарний запас	Рівень команди
1	2	3	4	5
UI|UX	Достатній	Наявний	Немає	Середній
Portal	Достатній	Немає	Низький	Середній
Migration	Недостатній	Наявний	Низький	Середній
Services	Недостатній	Наявний	Немає	Високий

Також на основі інформації представленої в табл. 3.16. – 3.19, формуються початкові дані, необхідні для вирішення цієї задачі.

Перейдемо до демонстрації рішення задачі моніторингу проєкту у типовій аутсорсинговій IT-компанії, використовуючи програмну оболонку. Спершу додаємо новий профіль та заповнимо базу знань на основі попередньо сформованих шаблонів правил – рис. 3.24.

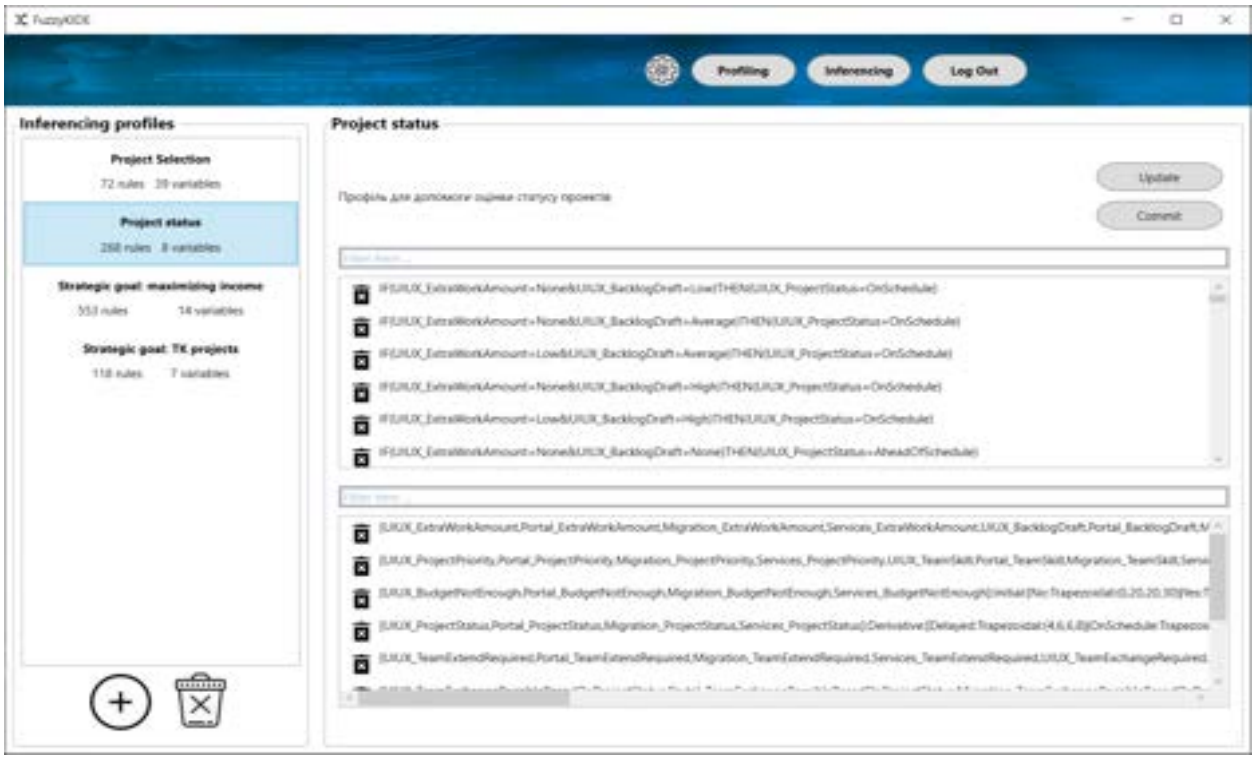


Рисунок 3.24 – Заповнена база знань для задачі оцінки статусу проєкту (Джерело – власна розробка)

Далі вкажемо початкові дані, сформовані на основі інформації представленої в табл. 3.16 – 3.19, та почнемо процес нечіткого виводу.

На рис. 3.25 наведено один з результатів нечіткого виводу для оцінки статусу проєкту до початку першого спринту розробки програмного продукту: *Services Start Negotiate Team Exchange = No (16.67) was enabled with confidence factor 1*.

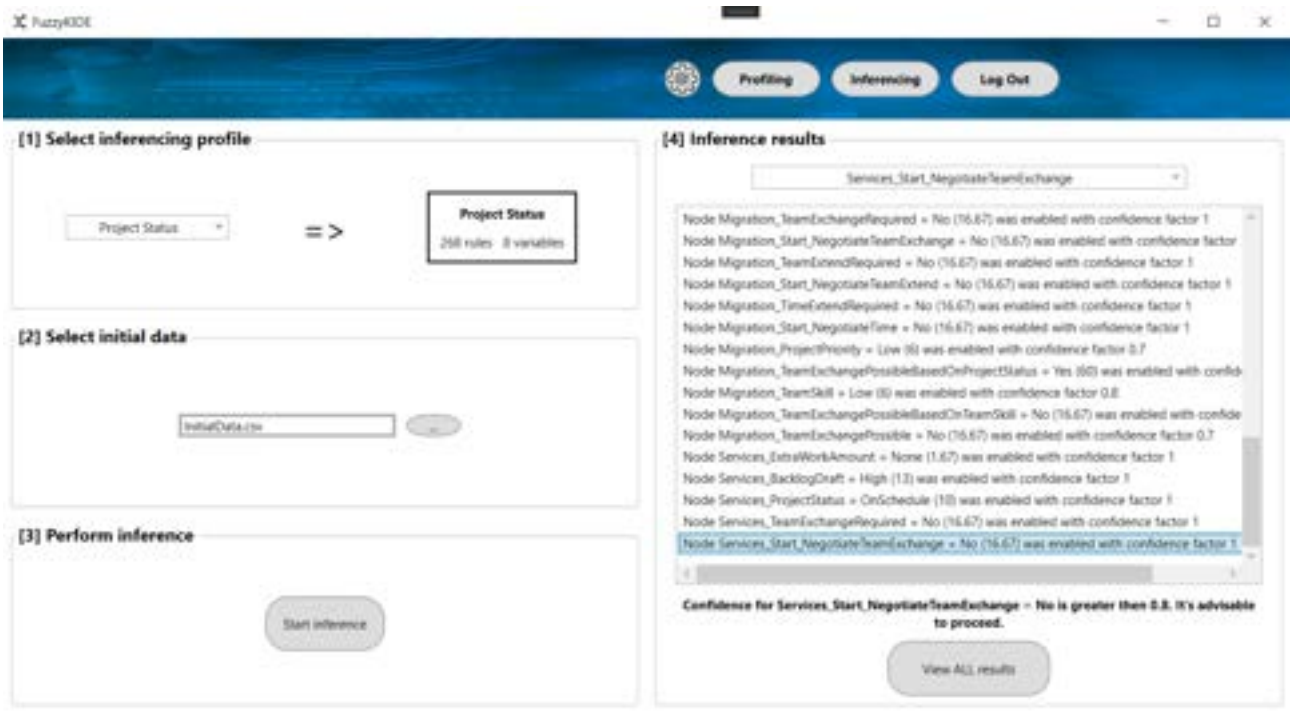


Рисунок 3.25 – Рішення задачі оцінки статусу проєкту: спринт 1

(Джерело – власна розробка)

Вузол *Services Start Negotiate Team Exchange = No (16.67)* було активовано с коефіцієнтом достовірності 1, що більше порогового значення за замовчуванням (0.8). Це свідчить про те, що експертна система не рекомендує дій щодо перетасування команду в ціллю вирівняти відставання команди Services. Це є логічним, адже на момент початку розробки ще немає підстав для рекомендацій щодо розширення строків виконання, бюджету, тощо, що експертна система і продемонструвала. Нечіткому значенню *No* відповідає чітке значення 16.67. Результат оцінки є багатофакторним, але в даному випадку інші результати не наведені, тому що вони дублюються.

На рис. 3.26 наведено результат нечіткого виводу для оцінки статусу проєкту приблизно в середині циклу розробки програмного проєкту, а саме до початку п'ятого спринту: *Migration Start Negotiate Budget = Yes (60) was enabled with confidence factor 0.8*.

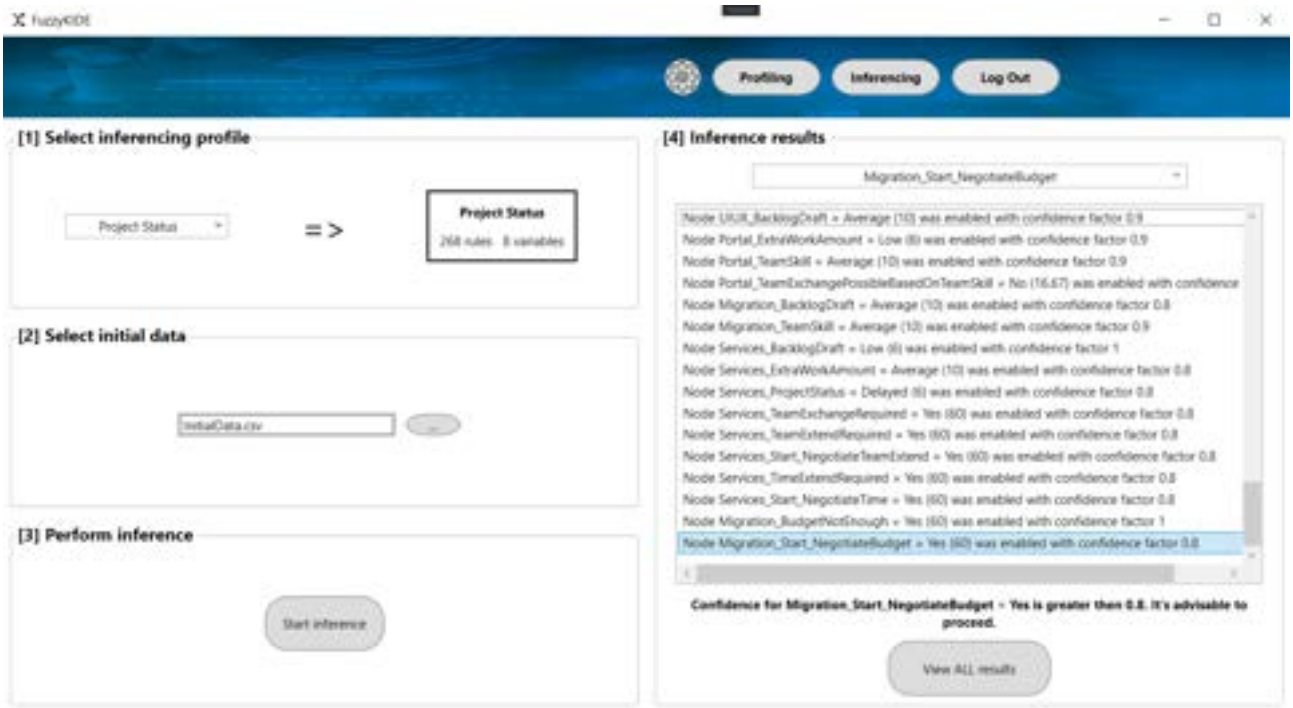


Рисунок 3.26 – Рішення задачі оцінки статусу проєкту: спринт 5
(Джерело – власна розробка)

Вузол *Migration Start Negotiate Budget = Yes (60)* було активовано с коефіцієнтом достовірності 0.8, що більше порогового значення за замовчуванням (0.8). Враховуючи нестачу бюджету для команди розробки Migration та неможливість перетасування команду з ціллю вирівнювання бюджету, експертна система рекомендує почати переговори щодо розширення бюджету для команди Migration. Нечіткому значенню *Yes* відповідає чітке значення 60. Як і в попередньому випадку результат є багатофакторним, але експертна система не дає рекомендацій щодо інших команд, тому усі інші результати не демонструються.

На рис. 3.27 та 3.28 можемо побачити результат нечіткого виводу для оцінки статусу проєкту ближче до кінця циклу розробки програмного продукту, а саме до початку дев’ятого спринту:

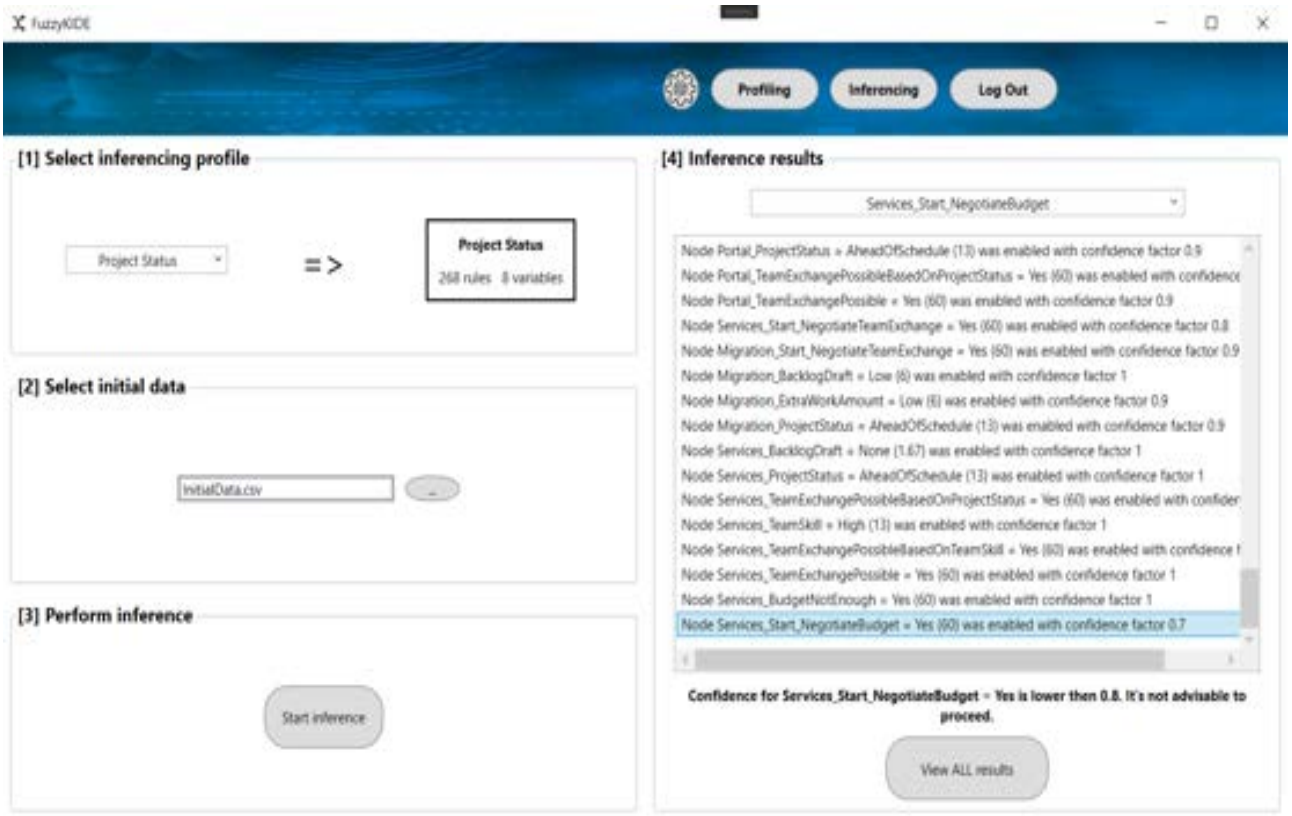


Рисунок 3.27 – Рішення задачі оцінки статусу проєкту: спринт 9

(Джерело –власна розробка)

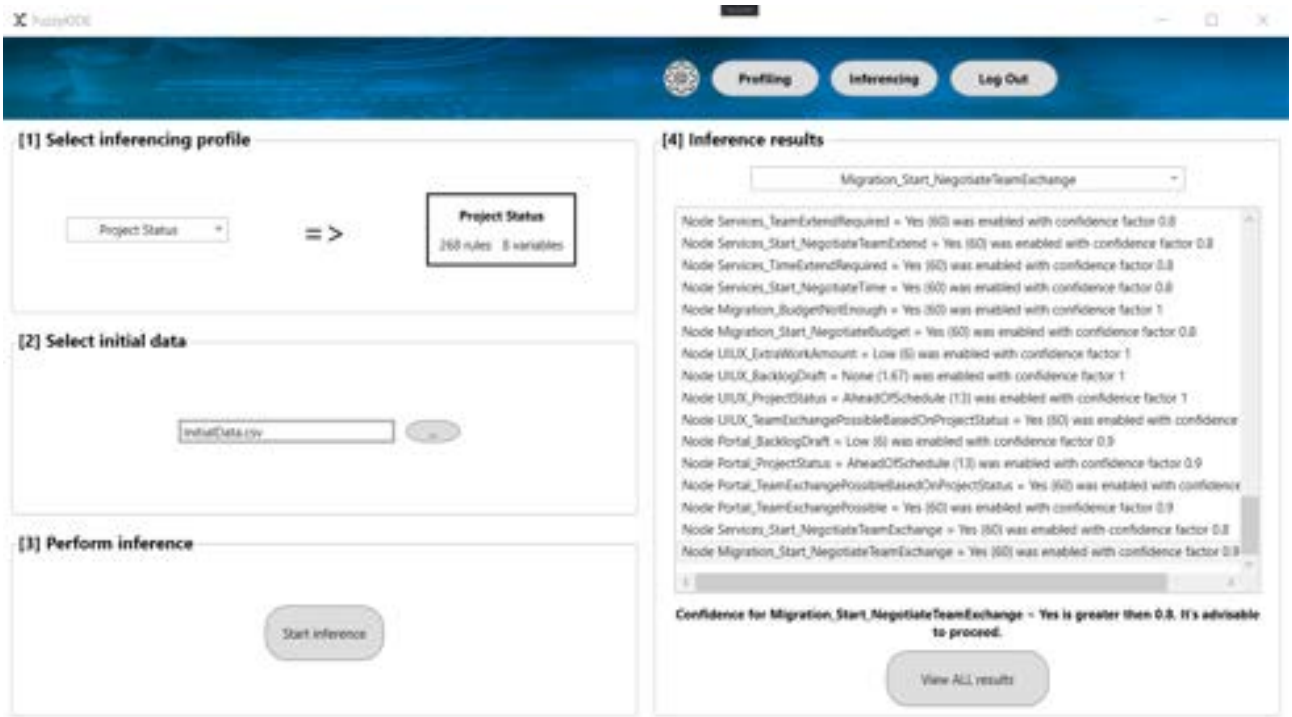


Рисунок 3.28 – Рішення задачі оцінки статусу проєкту: спринт 9

(Джерело –власна розробка)

- *Services Start Negotiate Budget = Yes (60) was enabled with confidence factor 0.8;*
- *Migration Start Negotiate Team Exchange = Yes (60) was enabled with confidence factor 0.9.*

Вузол *Services Start Negotiate Budget = Yes (60)* було активовано с коефіцієнтом достовірності 0.8, що більше порогового значення за замовчуванням (0.8). Також вузол *Migration Start Negotiate Team Exchange = Yes (60)* було активовано с коефіцієнтом достовірності 0.9 що дорівнює пороговому значенню за замовчуванням (0.8). На цьому прикладі можемо бачити багатофакторність результатів в дії, адже по перше, рекомендується почати переговори щодо розширення бюджету для команди розробки Services, а вдруге почати процес перетасування команд з ціллю вирівнювання бюджету для команди Migration. Якщо придивитися до ходу прийняття рішень експертної системи, то можна дізнатися, що таке рішення було прийнято, адже у команди Portal є можливість віддати розробників команди без ризику відставання для команди в цілому. В обох результатах нечіткому значенню *Yes* відповідає чітке значення 60. Як і в попередньому випадку усі інші результати не дають рекомендацій та не демонструються.

3.4 Аналіз результатів впровадження та обгрунтування ефективності застосування інноваційної технології – експертної системи FuzzyKIDE – в IT-сфері

Впровадження типової оболонки для розв’язання *задачі оцінки статусу портфоліо проєктів* з розділу 3.3 було апробовано на даних, близьких до реальних, та зроблено ретроспективно. Дані щодо процесу виконання проєктів

та відповідні економічні показники було сформовано на основі історичних даних, які залишилися після завершення портфоліо проєктів [44].

Почнемо з розрахунку абсолютних економічних показників. Для того щоб зробити висновок про успішність запропонованого методу управління проєктами нам потрібно вміти вимірювати його продуктивності. Критерії вартості, часу та якості розглядаються Шрнхаром [96] та Баккаріні [31] як внутрішні міри ефективності управління проєктами. Два з цих трьох критеріїв, тривалість та вартість виконання проєкту, можна порахувати як абсолютні величини, при цьому менші значення відповідають більш ефективному управлінню проєкта.

Тривалість виконання проєкту можна порахувати як кількість проведених спринтів від дати початку проєкту до дати його завершення.

Щоб формалізувати неявні припущення, що лежать в основі обчислення вартості виконання проєкту, можемо визначити зусилля E_R як результат використання ресурсу з деякою інтенсивністю R_I через задану тривалість Δt. Будемо використовувати загальнолінійне наближення для того щоб порахувати зусилля:

$$E_R = R_I \Delta t \ .$$

Можна порівняти ці зусилля, перетворивши їх на витрати за допомогою норми витрат C_R, яку можна отримати із структури розподілу ресурсів [89]. Одиницями цих конкретних ставок вартості є одиниці валюти за одиницю зусилля, а нижчий індекс R нагадує нам, що ставка витрат змінюється залежно від кожного окремого ресурсу. Підсумовуючи:

$$C = E_R C_R \ ,$$

який належним чином вводить загальну вартість C, яка стає конкретною при розрахунках зароблені вартості.

Тепер перейдемо до розрахунків комбінованих економічних показників. Аналіз зробленої вартості поєднує в собі три елементи бюджету, графіку та обсягу, використовуючи вартість як загальне середовище обміну. Таким чином, одиниця первинної фінансової валюти проєкту (наприклад, долари, фунти, євро) стає одиницею для всіх показників зробленої вартості. Тому можна порівняти різні виміри, оскільки вони мають спільну основу.

Система зробленої вартості включає обсяг та інтегрує його з витратами та графіком. По-перше, необхідно визначати вартість повністю завершеного проєкту у контексті витрат, які були передбачені та узгоджені у планах проєкту. Лише коли виконано певний обсяг завдання, проєкт заробляє вартість, і обсяг цієї вартості визначається витратами, які були передбачені бюджетом. Далі аналіз зробленої вартості включає графік у цю загальну основу порівняння, запитуючи, скільки витрат мало відбутися, відповідно до графіка проєкту.

В основному є три терміни, що ідентифікують зароблену вартість:

- Запланована вартість (C_s, scheduled cost) – затверджений бюджет, призначений на планові роботи.
- Фактична вартість (C_a, actual cost) – фактична вартість виконаної роботи – це сукупність прямих та непрямих витрат, понесених на виконання робіт за даною діяльністю протягом виконання проєкту.
- Зароблена вартість (C_b, budgeted cost) – передбачена в бюджеті вартість виконаної роботи. Вона пов’язує початкові заплановані витрати на проєкт та швидкість, з якою команда завершує роботу над проєктом.

Різниця між бюджетною вартістю та двома іншими витратами визначає, чи проєкт триває, випереджає чи відстає від бюджету чи графіка:

- Різниця у вартості $\Delta C_a = C_b - C_a$, коли $\Delta C_a < 0$ проєкт перевищує бюджет.
- Різниця в розкладі $\Delta C_s = C_b - C_s$, коли $\Delta C_s < 0$ проєкт відстає від графіка.

Можемо сформувати стандартні показники ефективності зароблені вартості, порівнюючи зароблену вартість з кожною з двох інших витрат у триплеті:

- Коефіцієнт витрат $F_a = \frac{C_a}{C_b} = 1 - \frac{\Delta C_a}{C_b}$,
- Коефіцієнт розкладу $F_s = \frac{C_s}{C_b} = 1 - \frac{\Delta C_s}{C_b}$.

Проєкт виконується за графіком та бюджетом, коли $F_a = F_s = 1$. Проєкт випереджає бюджет або графік, коли фактори менше одиниці.

Коефіцієнт виконання графіку витрат:

$$\Phi = F_a + F_s - 1 + \sqrt{\frac{1}{2}(F_a - F_s)^2}$$

Віднімання 1 дозволяє комбінованому коефіцієнту дорівнювати 1 (як і окремим факторам), якщо проєкт виконується як за графіком, так і за бюджетом. Квадратний корінь різниці між факторами, зведений у квадрат, додається до суми, так що ефект одного високого індивідуального фактора не може бути прихований низьким числом в іншому факторі.

Тепер перейдемо до розрахунку відносних економічних показників. Успіх управління проєктами фокусується на процесі проєкту та, зокрема, на успішному виконанні цілей щодо витрат, часу та якості. Синонімом успіху є ефективність, тобто ступінь досягнення цілей [33]. Проєкти формуються для досягнення цілей, а успіх вимірюється тим, наскільки ці цілі були досягнуті.

Час R_t – успіх можна виміряти термінами дотримання розкладу [72, 76, 85, 108]. Час як показник успіху можна виміряти з точки зору перевищення / недотримання графіка як відсоток від початкового плану [73].

Витрати R_c – успіх можна виміряти термінами виконання бюджету [72, 76, 85, 108]. Витрати як показник успіху можна виміряти як перевищення / недотримання витрат як відсоток від початкового бюджету [73].

Якість R_q – успіх можна виміряти з точки зору відповідності функціональним та технічним специфікаціям [32, 76, 108]. Іншими словами, “проєкт повинен виробляти те, що він сказав, що буде виробляти” [23]. Успіх технічних показників залежить від того, наскільки були досягнуті технічні вимоги, зазначені на початку етапу виконання [48].

Перед проведенням розрахунків розглянемо історичні дані, на основі яких ці розрахунки будуть проводитись. Наявні історичні дані щодо виконання цього проєкту базуються на командному розподілі та тривалості проєкту. Стосовно тривалості проєкту, то проєкт було заплановано на 5 місяців (10 спринтів), а виконано за 6 місяців (12 спринтів), з яких останній місяць було витрачено на дороблення проєкту команди Migration. Приблизні наявні дані стосовно командного розподілу продемонстровано у табл. 3.20 та 3.21.

Таблиця 3.20 – Місячна вартість праці спеціаліста за рівнем (Джерело – власна розробка)

Junior	**Middle**	**Senior**
1	2	3
1000$	2000$	3000$

Таблиця 3.21 – Розподіл спеціалістів за командами (Джерело –власна розробка)

Services	**UI|UX**	**Portal**	**Migration**
1	2	3	4
2 juniors 1 middle 1 senior	2 middles 2 seniors	2 middles 2 seniors	2 juniors 1 middle

Коефіцієнти вартості для історичних даних були надані уже розрахованими, але для чіткості проведемо ці розрахунки ще раз. Це допоможе у подальших розрахунках аналогічних коефіцієнтів для ситуації з впровадженням програмної оболонки.

Запланована вартість C_s є бюджетом на виділених спеціалістів впродовж запланованого терміну виконання проєкту, а саме 5 місяців:

C_s(Services) = (2 * 1000 + 1 * 2000 + 1 * 3000) * 5 = 35 000$

C_s(UI|UX) = (2 * 2000 + 2 * 3000) * 5 = 50 000$

C_s(Portal) = (2 * 2000 + 2 * 3000) * 5 = 50 000$

C_s(Migration) = (2 * 1000 + 1 * 2000) * 5 = 20 000$

$C_s = C_s$(Services) + C_s(UI|UX) + C_s(Portal) + C_s(UI|UX) = 155 000$

Фактична вартість C_a є фактичною вартістю роботи спеціалістів впродовж усього терміну виконання проєкту, а саме 6 місяців. Також будемо мати на увазі, що після спринту 5 до каманди Migration було додано одного Middle спеціаліста, а після спринту 9 до команди Services також було додано одного Middle спеціаліста. З урахуванням цієї інформації розрахуємо C_a:

C_a(Services) = (2 * 1000 * 5 + 1 * 3000 * 5) +

(1 * 2000 * 4.5 + 2 * 2000 * 0.5) = 36 000$

C_a(UI|UX) = (2 * 2000 + 2 * 3000) * 5 = 50 000$

C_a(Portal) = (2 * 2000 + 2 * 3000) * 5 = 50 000$

C_a(Migration) = 2 * 1000 * 5 + 1 * 2000 * 2.5 + 2 * 2000 * 2.5 +

(2 * 1000 + 2 * 2000) * 1 = 31 000$

C_a= C_a(Services) + C_a(UI|UX) + C_a(Portal) + C_a(UI|UX) = 167 000$

Зароблена вартість C_b є вартість вчасно розроблених проєктів. Тобто у розглядаємих даних це проєкти Services, UI|UX та Portal. Вартість незавершеного проєкту Migration для цього показника не враховується:

$C_b = C_a - C_a$(Migration) = 136 000$.

Розраховано всі дані, необхідні для підрахунку коефіцієнту виконання графіку витрат: $Ф = 1.416$. Значення більше одиниці свідчить про відставання розглядаємого набору проєктів.

Для повноти також проведемо розрахунки відносних коефіцієнтів. Успіх набору проєктів з точки зору часу можна розрахувати як відношення дійсно

витраченого часу до запланованого часу виконання цього набору проєктів (у місяцях): $R_t = (\frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{6}{5}) \div 4 = 1.05$.

Успіх набору проєктів з точки зору витрат можна розрахувати як відношення показника C_a до показника C_s (у $): $R_c = C_a \div C_s = 1.077$.

Успіх набору проєктів з точки зору якості можна розрахувати як відношення кількості вчасно розробленого функціоналу до запланованої кількості функціоналу. У даному прикладі розроблений функціонал можна розглядати з точку зори завершення / незавершення підпроєктів. Так як вчасно було завершено три з чотирьох проєктів, то: $R_q = 3 \div 4 = 0.75$.

Значення R_t більше одиниці свідчить про негативне відхилення від успішності за фактором часу. Значення R_c більше одиниці свідчить про негативне відхилення від успішності за фактором якості. Значення R_q більше одиниці свідчить про негативне відхилення від успішності за фактором витрат.

Тепер розрахуємо аналогічні коефіцієнти з урахуванням впровадження рекомендацій, запропонованих експертною системою. Але маємо на увазі, що так як надані дані є історичними і повторного виконання проєкту не відбувалося, то впровадження експертної системи є модельним. Розрахунки вартостей проводяться аналогічно з попередньо розглянутими прикладами, тому їх деталі опускаються.

Також як і раніше врахуємо, що у команду Migration було додано одного Middle спеціаліста після спринту 5, а в команду Services також одного Middle спеціаліста після спринту 9. Крім цього експертна система рекомендувала почати процес перетасування спеціалістів між командами з ціллю вирівняти командний професійний рівень команди Migration. Спам’ятаємо що це досягається за рахунок можливості команди Portal віддати одного зі спеціалістів без ризику відставання для команди.

Розрахунок запланованої вартості C_s не відрізняється від аналогічного розрахунку історичних даних, тому перейдемо одразу до результатів розрахунків фактичної вартості C_a:

C_a(Services) = 36 000$

C_a(UI|UX) = 50 000$

C_a(Portal) = 49 000$

C_a(Migration) = 26 000$

$C_a = C_a$(Services) + C_a(UI|UX) + C_a(Portal) + C_a(UI|UX) = 161 000$

З урахуванням рекомендацій експертної системи планується що усі чотири підпроєкти завершаться вчасно. В такому разі значення бюджетної вартості C_b не буде відрізнятися від значення уже розрахованої дійсної вартості C_a: C_b=C_a=161 000$.

Маючи всі необхідні дані, розрахуємо коефіцієнт виконання графіку витрат: Ф = 0.988. Значення близьке до одиниці свідчить про вчасне завершення розглядаємого набору проєктів без перевитрат.

Аналогічно з історичними даними проведемо розрахунки відносних економічних показників: $R_t = (\frac{5}{5} + \frac{5}{5} + \frac{5}{5} + \frac{5}{5}) \div 4 = 1$, $R_c = C_a \div C_s = 1.038$, $R_q = 4 \div 4 = 1$.

Значення R_t рівне одиниці свідчить про відсутність відхилення від успішності за фактором часу. Значення R_c більше одиниці свідчить про негативне відхилення від успішності за фактором якості. Значення R_q рівне одиниці свідчить про відсутність відхилення від успішності за фактором витрат.

Перейдемо до порівняння результатів оцінки відносних та комбінованих коефіцієнтів успішності історичних даних та даних у результаті модельованого впровадження експертної системи. Почнемо з відносних коефіцієнтів якості. Як

можемо бачити впровадження експертної системи скоріше за все призвело би до вчасного завершення набору проєктів, про що свідчать коефіцієнти якості за часом R_t та якістю R_q рівні 1. У свою чергу, для історичних даних ці самі коефіцієнти свідчать про негативні відхилення від якості. Більше цікаво спостерігати ситуацію з коефіцієнтом якості за витратами R_c. Для обох наборів даних можемо спостерігати негативе відхилення, але у разі впровадження експертної системи маємо значення 1.038, що все таки трохи менше за історичні дані 1.077, що свідчить про менше відхилення у разі використання експертної системи.

Тепер перейдемо до коефіцієнту виконання графіку витрат Ф. Тут ситуація більш однозначна. Для історичних даних значення цього коефіцієнта 1.416 свідчить про негативні відхилення від запланованого процесу та відставання набору проєктів від плану виконання, що також підтверджується і відносними коефіцієнтами. Натомість значення 0.988 при використанні експертної системи свідчить хоть і про незначні, але все таки позитивні відхилення у процесі та про відсутність відставання від плану виконання.

На остаток розглянемо цікаві результати поза межами відносних та комбінованих коефіцієнтів. Перший результат це те що, експертна система рекомендує почати процес переговорів щодо розширення бюджету з замовником якраз у момент, коли це найбільш необхідно, а саме після розширення команди Migration після спринту 5, та у команди Services після спринту 9. У разі упущення цього моменту є вірогідність що цю різницю у вартості роботи спеціалістів аутсорс IT-компанія виплатить власноруч, а у разі вчасного вирішення цього питання ці збитки понесе компанія замовник продукту. Другий результат є більш очевидним, адже стосується вчасного завершення проєкту Migration. В залежності від домовленості між аутсорс IT-компанією та компанією замовником, одна з них понесе збитки у розмірі вартості роботи спеціалістів на час, необхідний для завершення підпроєкту Migration, що не є бажаним результатом. У разі використання експертної

системи та вчасного перетасування команд з ціллю вирівняним професійний рівень команди Migration такого результату на передбачається.

Впровадження типової оболонки для вирішення *задачі максимізації прибутку* з розділу 3.3.2 було апробовано на даних близьких до реальних. На основі історичних даних було сформовано модельний річний звіт щодо розрахунку загальної виручки.

З ціллю зробити висновок про успішність запропонованого методу необхідно знайти спосіб вимірювання його продуктивності. Вимірювання продуктивності – це процес кількісної оцінки дій, що призводять до звершення, де вимірювання – це процес кількісної оцінки ефективності та результативності цих дій [79]. В даному прикладі вмістно розглядати критерій чистого прибутку як міри ефективності виконання задачі максимізації прибутку компанії. Далі розглянемо детальніше процес розрахунку чистого прибутку.

Спершу розглянемо основні показники, які приймаються участь у розрахунку чистого прибутку. У табл. 3.22 наведено загальний опис цих показників.

Таблиця 3.22 – Показники розрахунку прибутку (Джерело – власна розробка)

Назва	**Оригінальна назва**	**Опис**
1	2	3
Чистий прибуток	Net revenue	Різниця між прибутком фірми та усіма витратами. З точки зору бухгалтерського обліку, чистий прибуток - це різниця між валовим прибутком та витратами, пов'язаними з управлінням фірмою.
Загальна виручка	Total revenue	Загальна сума продажів компанії та інші джерела прибутку. Важливо зазначити, що дохід відрізняється від чистого прибутку, так як не враховує витрати.
Валовий дохід	Gross revenue	Дохід за фактурною вартістю, отриманий за товари та послуги протягом певного періоду часу.
Вартість проданих товарів	Cost of goods sold (COGS)	Балансова вартість товарів або послуг, проданих протягом певного періоду.
Операційні витрати	Operating expenses	Сума, сплачена за утримання активів або витрати на ведення бізнесу, за винятком амортизації.

Почнемо з розрахунку валового доходу (Gross revenue):

Gross revenue = Total revenue – COGS,

де значення загальної виручки (Total revenue) можна отримати сумуючи плановий річний прибуток від проєктів взятих у розробку, а вартість проданих товарів (COGS) у нашому випадку є вартістю послуг, тобто сумарною вартістю компенсацій команд, працюючих над проєктами, взятими у розробку.

Маючи значення загальної виручки є можливість розрахувати значення чистого прибутку (Net revenue):

Net revenue = Gross revenue – Operating expenses,

де операційні витрати (Operating expenses) у нашому випадку є витратами на рекламні кампанії та на активи у вигляді офісного приміщення та адміністративного персоналу.

Таким чином на основі показників, для яких наявні дані, можна сформулювати наступний спосіб розрахунку чистого прибутку:

Net revenue = Total revenue – COGS – Operating expenses.

Продовжимо огляд проєкту, описаного у розділі 3.3.2. Наявні історичні дані базуються на річному розрізі даних приближених до реальних, а саме:

- Плановий річний прибуток від потенційних проєктів;
- Місячні компенсації поточних команд та адміністративного персоналу;
- Річні витрати на підтримку активного офісу;
- Місячні витрати на підтримку активних рекламних кампаній.

Приблизні наявні дані щодо планових річних прибутків від проєктів можна переглянути в табл. 3.23 та 3.24.

Таблиця 3.23 – Плануємі річні прибутки від потенційних проєктів (Джерело – власна розробка)

Genesis	**Crowding**	**Firantis**	**Exploration**	**Hymera**
1	2	3	4	5
350 000 $	250 000 $	285 000 $	90 000 $	320 000 $

Таблиця 3.24 – Плануємі річні прибутки від поточних проєктів (Джерело –власна розробка)

CycleDuo	**Templater**	**Avion (залишок)**	**Howl (залишок)**	**Converge (залишок)**
1	2	3	4	5
150 000 $	90 000 $	50 000 $	45 000 $	35 000 $

Місячні компенсації команд продемонстровано в табл. 3.25.

Таблиця 3.25 – Компенсації поточних команд (Джерело – власна розробка)

CycleDuo	**Templater**	**Avion (залишок)**	**Howl (залишок)**	**Converge (залишок)**	**Emerald (нова команда)**
1	2	3	4	5	6
7 000 $	4 000 $	12 000 $	8 000 $	3 000 $	5 000 $

Річні витрати на підтримку активів у вигляді офісу та адміністративного персоналу надано в табл. 3.26.

Таблиця 3.26 – Витрати на підтримку активів (Джерело – власна розробка)

Витрати на адміністративний персонал (місячні)	**Витрати на офіс (місячні)**
1	2
15 000 $	18 000 $

Місячні витрати на активні рекламні кампанії надано в табл. 3.27.

Таблиця 3.27 – Річні витрати на підтримку рекламних кампаній (Джерело – власна розробка)

Соціальні мережі	**Конференції**	**Університети**	**Зовнішня реклама**
1	2	3	4
12 000 $	90 000 $	50 000 $	15 000 $

Значення річного чистого прибутку було надано розрахованим, але для візуалізації проведемо цей розрахунок ще раз. Це допоможе провести аналогічний розрахунок у випадку впровадження програмної оболонки. Перед проведенням розрахунку скорочено опишемо реальну ситуацію в прийнятті рішень на основі даних, описаних в вище:

- Для виконання проєкту Genesis було обрано команду Howl;
- Для виконання проєкту Hymera було обрано команду Converge;
- Для виконання проєкту Crowding було обрано команду Avion;
- Для виконання проєкту Firantis було найнято нову команду Emerald;
- Проєкт Exploration не було взято до розробки;
- Було прийнято рішення не відмовлятися від офісу;
- Було скорочено розмір адміністративного персоналу;
- Було розширено бюджет на рекламну кампанію в соціальних мережах;
- Було розширено бюджет на рекламну кампанію в університетах;
- Було залишено без змін бюджет на конференції;
- Було прийняте рішення відмовитися від зовнішньої реклами.

Спочатку розрахуємо заплановану загальну виручку на основі плануємих річних прибутків від проєктів. Маємо на увазі, що вже існуючі проєкти також враховуються у обчисленнях незалежно від строку їх закінчення. Також цікавим моментом є провал проєкту Crowding командою Avion через комбінацію високого ризику взаємодії з замовником Mazzle та командою Avion. Таким чином проєкт Crowding у розрахунках не враховується:

$$\textit{Total revenue} = 12 * 1000 * (150 + 90 + 50 + 45 + 35) +$$
$$12 * 1000 * (350 + 285 + 320) = 1\,640\,000\ \$.$$

Тепер розрахуємо річну вартість проданих товарів, що у цьому прикладі є суммою компенсацій команд задіяних у розробці. Команди, задіяні у розробці поточних проєктів, які ще не закінчилися також враховуються у цьому розрахунку:

COGS = 12 * 1000 * (7 + 4 + 12 + 8 + 3 + 5) = 468 000 $.

Залишилося розрахувати операційні витрати, які складаються з витрат на рекламні кампанії, підтримку офісу та адміністративного персоналу:

Operating expenses = 12 * 1000 * (18 + 15) +

1000 * (12 + 50 + 90 + 15) = 563 000 $.

Отримано усі показники необхідні для розрахунку чистого прибутку:

Net revenue = 1 640 000 – 468 000 – 563 000 = 609 000 $.

Далі проведемо аналогічні розрахунки чистого прибутку з урахуванням впровадження рекомендацій експертної системи. Але треба мати на увазі, що так як дані є історичними, то впровадження експертної системи є модельним. Спершу згрупуємо рішення запропоновані експертною системою:

- Для виконання проєкту Genesis пропонується обрати команду Howl;
- Для виконання проєкту Hymera пропонується обрати команду Converge;
- Для виконання проєкту Crowding пропонується найняти нову команду Emerald;
- Для виконання проєкту Firantis пропонується обрати команду Avion;
- Проєкт Exploration пропонується не брати до розробки;
- Пропонується відмовитися від офісу;
- Пропонується скоротити розмір адміністративного персоналу;
- Пропонується розширити бюджет на рекламну кампанію в соціальних мережах;
- Пропонується залишити без змін бюджет на рекламну кампанію в університетах;
- Пропонується відмовитися від рекламної кампанії посередством конференцій;
- Пропонується відмовитися від зовнішньої реклами.

Розрахуємо загальну виручку на основі плануємих річних прибутків від наявних та потенційних проєктів. На відміну від дійсних історичних даних, проєкту Crowding відповідає команда з меншим ризиком співпраці, що дозволяє припускати більшу вірошідність успішного завершення проєкту. Таким чином у цьому розрахунку проєкт Crowding враховується:

Total revenue = 12 * 1000 * (150 + 90 + 50 + 45 + 35) +

12 * 1000 * (350 + 250 + 285 + 320) = 1 890 000 $.

Розрахунок суми компенсацій команд задіяних у розробці не відрізняється від розрахунку у історичних даних, адже у розробці задіяні ті самі команди та складає *COGS* = 468 000 $. Таким чином одразу перейдемо до розрахунку операційних витрат. Відмінність від історичних даних полягає у пропозиції відмовитися від активного офісу, відмовитися від рекламної кампанії посередством конференцій та не змінювати бюджет на університетські рекламні кампанії. Зменшуємо бюджет на університетські рекламні кампанії в порівнянні з історичними даними з 50 000 $ до 30 000 $, що є розміром цього бюджету до розширення. Розширюємо бюджет на рекламну кампанію у соціальних мережах приблизно у тих самих пропорціях з 12 000 $ до 20 000 $. Також відмовляючись від рекламної кампанії посередством конференцій та підтримки офісу загалом маємо:

Operating expenses = 12 * 1000 * 15 + 1000 * (20 + 30) = 230 000 $.

Отримано усі показники необхідні для розрахунку чистого прибутку:

Net revenue = 1 890 000 – 468 000 – 230 000 = 1 192 000 $.

Розглянемо у яких місцях співпадають та відрізняються історичні рішення та запропоновані експертної системою та продемонструємо впливи цих різниць. Коротко резюмуємо різниці у прийнятті рішень:

- Для виконання проєкту Crowding пропонується найняти нову команду Emerald на відміну від команди Avion;
- Для виконання проєкту Firantis пропонується обрати команду Avion на відміну від команди Emerald;
- Пропонується відмовитися від офісу;

- Пропонується розширити бюджет на рекламну кампанію в соціальних мережах на відміну від історичних даних, які залишили цей бюджет без змін;
- Пропонується залишити без змін бюджет на рекламну кампанію в університетах на відміну від історичних даних, які збільшили цей бюджет;
- Пропонується відмовитися від рекламної кампанії посередством конференцій на відміну від історичних даних, які залишили цей бюджет без змін.

Інші рішення експертної системи аналогічні рішенням з історичних даних. Першою основною відмінністю є відмінність у пошуку команди для проєкту Crowding. Експертна система проаналізувала ризик співпраці з замовником та наявними командами та прийшла до рішення, що через комбінацію високих ризиків є менш ризиковим найняти для виконання проєкту нову команду, ніж призначати одну з наявних. Таким чином більша вірогідність успішно завершити проєкт та уникнути ситуації продемонстрованої у історичних даних з провалом проєкту Crowding через несумісність з командою Avion. Враховуючи цю інформацію, команді Avion було доручено роботу над проєктом Firantis.

Наступним моментом є пропозиція щодо відмови від активного офісу. Таке рішення було прийнято проаналізувавши ризики співпраці з поточними командами та отримавши невисокий загальний ризик. Також було проаналізовано корисність поточних рекламних кампаній. Було розширено бюджет на кампанію в соціальних мережах, що з одного боку розширює витрати, але через високу корисність та невеликі вкладення створює найбільш сприятливе середовище для пошуку нових команд та замовників. Це у перспективі призводить до більш стрімкого розширення компанії та збільшує вірогідність знайти замовника, що призведе до збільшення значення загального прибутку (Total revenue). Через середній рівень корисності та витрат було прийнято рішення залишити бюджет університетських рекламних кампаній без

змін. Проаналізувавши низький рівень користі та великі витрати на кампанію посередством конференцій експертною системою було прийняте однозначне рішення відмови від цього типу кампанії.

Можемо спостерігати вплив цих рішень на значення загального прибутку та операційних витрат. Загальний прибуток зріс на 250 000 $. У свою чергу, операційні витрати зменшилися на 333 000 $. І загалом з урахуванням впровадження експертної системи можемо бачити зростання річного чистого прибутку (Net revenue) на 583 000 $.

Висновки до розділу 3

Представлено результати проведення серії експериментів, що демонструють потужність та ефективність програмної оболонки FuzzyKIDE у вирішенні завдань управління проєктами. Наведено процес ретроспективного аналізу історичних даних та даних, отриманих в результаті впровадження програмної оболонки FuzzyKIDE.

Завдяки FuzzyKIDE стало можливим виконати аналіз, порівняння та відбір найбільш прийнятних проєктів, що відповідають потребам компанії. Експеримент підтвердив, що використання програмної оболонки значно сприяє об'єктивному та точному відбору проєктів, що мають потенціал для успішної реалізації.

Проведено серію експериментів для оцінки стану реалізації проєкту та виявлення можливих недоліків. Застосування програмної оболонки FuzzyKIDE дозволило систематизувати та аналізувати дані про стан проекту за різними аспектами. Результати експерименту підтвердили, що використання FuzzyKIDE значно сприяє визначенню обґрунтованих висновків та прийняттю відповідних корекційних заходів для покращення ефективності виконання проєктів.

Проведено експерименти для вирішення стратегічних завдань управління компанією. Результати досліджень довели, що використання програмної оболонки FuzzyKIDE сприяє досягненню вищої професійної спрямованості компанії та максимізації прибутку шляхом забезпечення обґрунтованості та оптимальності стратегічних рішень. Експерименти є прикладами успішного використання програмної оболонки FuzzyKIDE та підтверджують її ефективність у рішенні специфічних завдань управління проєктами. Результати експериментів слугують додатковою вагомою підтримкою для компаній, які планують впровадження FuzzyKIDE у своїх операціях.

Одним з потенційних напрямків розвитку для програмної оболонки є можливість створення метакомпоненти для збереження результатів ретроспективних аналізів, отриманих в процесі її використання, що може послужити подальшому уточненню процесу нечіткого виводу. Для розробки метакомпоненти необхідна продумана структура збереження результатів ретроспективного аналізу у базі прецедентів, яка б враховувала алгоритмічні особливості модифікованого механізму виводу та розробленої на цій основі семантичної мережі.

Результати розділу 3 висвітлено у таких працях автора, наведених у Додатку А: [1, 4, 5, 8].

ВИСНОВКИ

У дисертаційній роботі розроблено рекомендації щодо застосування інноваційних технологій в процесі управління аутсорсингових IT-компаній в частині формування інструментально-аналітичного забезпечення ефективного управління IT-проєктами та обґрунтування економічних рішень на основі їх експертної оцінки.

Запропоновні підходи до підвищення ефективності управління аутсорсинговими IT-проєктами полягають у використанні Agile технології, State-Gate фреймворку та теорії нечітких множин.

Основні наукові і практичні результати роботи полягають у наступному:

1. Дослідженням динаміки та особливостей розвитку вітчизняної та світової IT-галузі встановлено, що за останні кілька десятиліть IT-галузь пережила значний розвиток і стала однією з найбільш динамічно зростаючих галузей у світі. За даними провідних аналітичних компаній, зокрема IDC та Gartner, світовий ринок IT-послуг та продуктів продовжує зростати. В умовах пандемії COVID-19, IT-галузь демонструє велику резилієнтність і здатність адаптуватися до змінних умов ринку. В Україні також спостерігається стабільний розвиток IT-індустрії, що забезпечує значний внесок в економіку країни та створює робочі місця для тисяч людей. Українська IT-галузь відзначається динамічним розвитком. За оцінками Асоціації IT-підприємств України, у 2021 році обсяг ринку IT-послуг в Україні зріс на 20%, а кількість співробітників виробництва програмного забезпечення перевищило 200 тисяч. Крім того, за даними Global Outsourcing 100, Україна знаходиться серед лідерів світового ринку аутсорсингових послуг та є привабливою країною для зарубіжних інвесторів. Однією з основних особливостей розвитку IT-галузі є постійне зростання вимог до якості та швидкості розробки програмного забезпечення. Це призводить до використання новітніх технологій та методологій розробки, таких як Agile та DevOps, які дозволяють ефективно

вирішувати завдання в умовах швидкозмінного ринку та постійно зростаючих вимог користувачів.

2. Аналіз діяльності IT-компаній визначив ряд типових проблем, що виникають в процесах управління проєктами/портфелями проєктів. Зокрема, такими проблемами є відсутність чіткої формалізації процесів управління проєктами; недостатня документованість проєктів; неефективне використання ресурсів; недостатня комунікація між учасниками проєкту; нестача контролю за процесами виконання робіт; невдале використання програмного забезпечення, що підтримує управління проєктами; а також неспроможність підвищувати ефективність процесів управління проєктами. Для вирішення цих проблем необхідно використовувати сучасні підходи та технології управління проєктами, такі як Agile, Scrum, Kanban, які дозволяють підвищити ефективність управління проєктами та розв'язувати типові проблеми. Крім того, для досягнення успіху в управлінні проєктами важливо забезпечити належний рівень комунікації між учасниками проєкту, дотримуватися затверджених термінів та забезпечити належну координацію робіт. Таким чином, використання сучасних технологій та підходів в управлінні проєктами є ключовим для успішної діяльності IT-компаній, оскільки дозволяє підвищувати ефективність процесів та досягати поставлених цілей.

3. Аналіз сучасних підходів та інноваційних технологій управління проєктами/портфелями проєктів, що використовуються в практиці роботи IT-компаній, довів певне коло найбільш придатних та ефективних для розв’язання поставлених задач дослідження. Серед них Agile, Scrum, Kanban та Stage-Gate фреймворк. В якості базової інноваційної технології до розробки інструментально-аналітичного забезпечення обґрунтування й прийняття економічних рішень з управління портфелем IT-проєктів обрано Agile Stage-Gate фреймворк, що обумовлено наступним. Цей фреймворк поєднує в собі принципи Agile технології та Stage-Gate моделі управління проєктами, що дозволяє досягати кращих результатів у розробці IT-продуктів. Agile методологія забезпечує гнучкий і ітераційний підхід до розробки продукту, що

дозволяє швидко реагувати на зміни у вимогах та уникнути затримок у процесі розробки. Stage-Gate модель, у свою чергу, забезпечує систематичний і керований підхід до управління проєктами, що дозволяє ефективно впроваджувати технологічні інновації та підвищувати якість продукту. Таким чином, застосування Agile Stage-Gate фреймворку до управління портфелем IT-проєктів дозволяє забезпечити гнучкість та ефективність у процесі розробки, покращити якість продукту та підвищити задоволеність клієнтів. Крім того, використання цього фреймворку дозволяє зменшити ризики та витрати на управління проєктами, що є важливим фактором успіху в IT-галузі.

4. Запропоновано концептуальну модель обґрунтування й прийняття економічних рішень з управління проєктами в умовах невизначеного середовища і неточної інформації. Сутність концепції полягає у використанні апарату нечіткої логіки в загальному контурі прийняття рішень, який заснований на технології Agile Stage-Gate фреймворк, обраної в якості базової інноваційної технології. Порівняно з класичним підходом до використання Stage-Gate фреймворку, модифікація фреймворку дозволяє описувати правила прийняття рішень та правила переходу з етапу на етап у лексичному форматі, а також зменшує необхідність у великій кількості точних числових даних на користь лексичних даних. Унікальною особливістю запропонованої моделі є використання систем нечіткої логіки при переході від брам до наступних етапів, що забезпечує більш гнучкий підхід до управління проєктами. Додатково, в деяких випадках застосування систем нечіткої логіки може бути ітеративним в залежності від ітеративності самого етапу.

5. Розроблена інструментально-аналітична база підтримки прийняття економічних рішень з управління проєктами/портфелями проєктів, ядром якої є нечітка експертна система FuzzyKIDE. Модель бази знань системи – семантична мережа з нечіткими продукційними правилами. В ЕС використовується модифікований механізм нечіткого виводу, що відрізняє її від існуючих зразків. Завдяки запропонованій архітектурі з оптимізованим механізмом нечіткого виводу досягається зменшення навантаження на базу

даних, що призводить до прискорення процесу нечіткого виводу й підвищенню оперативності надання ЕС кінцевих результатів-рекомендацій. Сформована база знань ЕС FuzzyKIDE дозволяє розв’язувати типові задачі ІТ-компаній – пошуку проєктів й формування складу/структури портфоліо; оцінки статусу проєктів; завдань стратегічної орієнтації компаній (зокрема, професійного спрямування та максимізації прибутку) – на новому рівні. Головна відмінність від традиційних підходів – можливість здійснення комплексного економічного аналізу з врахуванням впливів факторів різної природи в умовах неповної/неточної інформації стосовно оточення ІТ-проєктів.

6. Проведені емпіричні дослідження підтвердили ефективність запропонованого науково-методичного підходу до розв’язання задач стратегічного управління ІТ-компанією. Конкретно, в результаті застосування математичного інструментарію FuzzyKIDE виконано комплексний аналіз, що дозволяє управлінцям моніторити досягнення цілей та приймати обґрунтовані рішення щодо розвитку компанії. Використання експертної системи FuzzyKIDE при формуванні складу і структури портфоліо проєктів виявилося дієвим методом прийняття економічних рішень у поточному періоді та на перспективу. Це досягається завдяки підвищенню рівня інформаційної підтримки управління, а саме, застосуванню баз даних та знань, що забезпечує якісний ввід інформації у систему, а також забезпечує максимально повну і достовірну інформацію для експертної оцінки. Підвищення оперативності та гнучкості прийнятих управлінських рішень досягнуто завдяки використанню систем нечіткої логіки при моніторингу поточного статусу ІТ-проєктів /портфоліо проєктів. Здійснений багатофакторний аналіз впливів внутрішнього та зовнішнього оточення ІТ-компаній дозволяє враховувати неоднорідність та нечіткість отриманої інформації та, як наслідок, зменшує можливість помилок у прийнятті рішень. Отже, результати емпіричних досліджень, проведених на базі розробленого науково-методичного підходу, свідчать про ефективність використання математичного інструментарію FuzzyKIDE в стратегічному управлінні ІТ-компанією, що забезпечує підвищення оперативності і гнучкості

прийнятих управлінських рішень, а також зменшення ризику та підвищення рівня прибутковості діяльності ІТ-компанії.

7. Розвинуто інформаційне забезпечення процесів розробки економічних рішень з управління ІТ-проєктами, що досягається за рахунок створення та подальшої актуалізації баз даних та знань нечіткої експертної системи FuzzyKIDE в ході експлуатації й емпіричних випробувань.

Апробація розробленого інструментально-аналітичного забезпечення обґрунтування й прийняття економічних рішень з управління аутсорсинговими ІТ-проєктами показала позитивні результати, довела реальний характер застосування апарату нечіткої логіки в діяльності ІТ-компаній.

Проведене дослідження в цілому підтвердило робочу гіпотезу дисертації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дудник О. О. Удосконалення процесів управління портфелем проєктів в IT-компаніях // Економічна кібернетика: теорія, практика та напрямки розвитку. 2017. С. 52-54.
2. Дудник О. О. Аналіз процесів управління портфелем проєктів в IT-компаніях // X Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем”. 2018.
3. Дудник О. О. Процес вибору виконавців IT-проєктів у портфоліо проєктів // Економічна кібернетика: теорія, практика та напрямки розвитку. 2018. С. 50-52.
4. Дудник О. О. Аналіз оптимальної структури бази знань для нечітких експертних систем // XI Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем”. 2019.
5. Дудник О. О. Переваги використання нечіткої логіки в процесі управління IT-проєктами // XIII Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем”. 2021.
6. Дудник О. О. Економіка та апарат нечітких експертних систем: тренди розвитку та застосування // Економічна кібернетика: теорія, практика та напрямки розвитку”. 2023. С. 69-73.
7. IT-експорт у 2020. За даними НБУ щодо платіжного балансу і зовнішньої торгівлі // BRDO [Електронний ресурс]. URL: https://brdo.com.ua/wp-content/uploads/2021/02/IT-eksport-u-2020-2.pdf (дата звернення: 21.08.2021).
8. Соколовська З. М., Дудник О. О. Експертні системи: теорія та прикладне застосування: [монографія]. Одеса: Екологія, 2022. 408 с.
9. Берштейн Л. С., Боженюк А. В. Нечеткие модели принятия решений: [монография]. Таганрог: Вид-во ТРТУЖ, 2001. 210 с.
10. Борисов А. Н., Алексеев А. В., Крумберг О. А. Модели принятия решений на основе лингвистической переменной. Рига: Зинатне, 1982. 256 с.

11. Борисов А. Н., Алексеев А. В., Меркурьева Г. В. Обработка нечеткой информации в системах принятия решений. Москва: Радио и связь, 1989. 304 с.
12. Борисов А. Н., Крумберг О. А., Федоров И. П. Принятие решений на основе нечетких моделей. Примеры использования. Рига: Зинатне, 1990. 184 с.
13. Жуковин В. Е. [та ін.]. Об одном подходе к задачам принятия решений с позиции теории нечетких множеств Рига: РПИ, 1980.С. 12–16.
14. Заде Л. А. Основы нового подхода к анализу сложных систем и процессов принятия решений Москва: Знание, 1974.С. 5–49.
15. Заде Л. А. Понятие лингвистической переменной и его применение к принятию приближенных решений. Москва: Мир, 1976. 165 с.
16. Кофман А. Введение в теорию нечетких множеств. Москва: Радио и связь, 2002. 432 с.
17. Нечеткие множества и теория возможностей – последние достижения под ред. Р. Р. Ягер, Москва: Радио и связь, 1986. 408 с.
18. Орловский С. А. Проблемы принятия решения при нечеткой исходной информации. Москва: Наука, 1981. 254 с.
19. Попов Э. В. Экспертные системы. Москва: Наука, 1988. 210 с.
20. Трифонов Ю. В., Плехонова А. Ф., Юрлов Ф. Ф. Выбор эффективных решений в экономике в условиях неопределенности: [монография]. Новгород: ННГУ, 1999. 140 с.
21. Уотермен Д. Руководство по экспертным системам. Москва: Мир, 1989. 388 с.
22. Частиков А. П., Леднева И. Ю. Использование байесовской сети при разработке экспертных систем с нечеткими знаниями. Краснодар: КубГТУ, 2000. 179 с.
23. A guide to the project management body of knowledge. Newtown Square: Project Management Institute, Inc., 1996.

24. Agápito A. D. O. [et al.]. Using multicriteria analysis and fuzzy logic for project portfolio management // Brazilian Journal of Operations & Production Management. 2019. № 2 (16). P. 347–357.
25. Aggarwal A., Thakur G. S. M. Techniques of Performance Appraisal - A Review // International Journal of Engineering and Advanced Technology. 2013.
26. Ambler S. W. The Disciplined Agile (DA) Framework [Електронний ресурс]. URL: http://www.disciplinedagiledelivery.com/introduction-to-dad (дата звернення: 22.08.2021).
27. Andrienko-Bentz O. Export-oriented segment of Ukraine's IT services market: Status quo and prospects // PwC Ukraine [Електронний ресурс]. URL: https://www.slideshare.net/RuslanSivoplyas/exportoriented-segment-of-ukraines-it-services-market2016 (дата звернення: 21.08.2021).
28. Anghel A. Outsourcing Company vs. Product Company // SlideShare [Електронний ресурс]. URL: https://www.slideshare.net/anghelalexandra/ outsourcing-company-vs-product-company (дата звернення: 22.08.2021).
29. Arnuphaptrairong T. Top ten lists of software project risks: Evidence from the literature survey 2011.P. 1–6.
30. Atkinson R., Crawford L., Ward S. Fundamental uncertainties in projects and the scope of project management // International Journal of Project Management. 2006. № 8 (24). P. 687–698.
31. Baccarini D. The Logical Framework Method for Defining Project Success // Project Management Journal. 1999. № 4 (30). P. 25–32.
32. Baker B. N., Murphy D. C., Fisher D. Factors affecting project success // Project management handbook. 1997. P. 902–919.
33. Belout A. Effects of human resource management on project effectiveness and success: Toward a new conceptual framework // International Journal of Project Management. 1998. № 1 (16). P. 21–26.
34. Bloch M., Blumberg S., Laartz J. Delivering large-scale IT projects on time, on budget, and on value // Harvard Business Review. 2012. P. 2–7.

35. Charette R. N. Software Engineering Risk Analysis and Management. USA: McGraw-Hill, Inc., 1989.
36. Choi J., Shapiro S. C. Efficient implementation of non-standard connectives and quantifiers in deductive reasoning systems // Institute of Electrical and Electronics Engineers (IEEE), 1992.P. 381–390.
37. Cooper R. Agile–Stage-Gate Hybrids // Research Technology Management. 2016. № 1 (59). P. 21–29.
38. Cooper R. G. The Stage-Gate® System: A Road Map from Idea to Launch – An Intro and Summary // GEMBA INSIGHT [Електронний ресурс]. URL: http://www.bobcooper.ca/images/files/articles/2/1-The-Stage-Gate-System-A-Roadmap-From-Idea-to-Launch.pdf (дата звернення: 22.08.2021).
39. Cooper R. G., Sommer A. F. The Agile–Stage-Gate Hybrid Model: A Promising New Approach and a New Research Opportunity // Journal of Product Innovation Management. 2016. № 5 (33). P. 513–526.
40. Corona E., Pani F. E. A Review of Lean-Kanban Approaches in the Software Development // WSEAS transactions on information science and applications. 2013. № 1 (10). P. 1–13.
41. The Development Impact of the War in Ukraine. Initial Projections // UNDP [Електронний ресурс]. URL: https://www.undp.org/sites/g/files/zskgke326/files/2022-03/Ukraine-Development-Impact-UNDP.pdf (дата звернення: 20.08.2022).
42. Digital resonance: the new factor influencing location attractiveness // Kearney [Електронний ресурс]. URL: https://www.kearney.com/digital-transformation/gsli/2019-full-report (дата звернення: 21.08.2021).
43. Dudnyk O., Sokolovska Z. Modern Technologies for Management of Outsourcing IT Projects // Business Inform. 2021. № 2. P. 62-72.
44. Dudnyk O., Sokolovska Z. Application of Fuzzy Expert Systems in IT Project Management // Project Management - New Trends and Applications, IntechOpen. 2022.

45. Dudnyk O., Sokolovska Z. Forecasting development trends in the information technology industry using fuzzy logic // Eastern-European Journal of Enterprise Technologies. 2023. № 1(13 (121). P. 74–85.
46. Holger S. Dr. Plattform-Ökonomie [Електронний ресурс]. URL: https://www.netzoekonom.de/plattform-oekonomie/ (дата звернення: 21.08.2021).
47. Elonen S., Artto K. A. Problems in managing internal development projects in multi-project environments // International Journal of Project Management. 2003. № 6 (21). P. 395–402.
48. Enoch C. N., Labuschagne L. Project portfolio management: using fuzzy logic to determine the contribution of portfolio components to organizational objectives // Project Management Institute. 2012.
49. Evaluate Offshore/Nearshore Countries for Outsourcing, Shared Services and Captives in EMEA, 2017 // Gartner [Електронний ресурс]. URL: https://www.gartner.com/en/documents/3762863 (дата звернення: 21.08.2021).
50. Freeman M., Beale P. Measuring project success // Project Management Journal. 1992. № 1 (23). P. 8–17.
51. Gartner Forecasts Worldwide IT Spending to Grow 3% in 2022 // Gartner [Електронний ресурс]. URL: https://www.gartner.com/en/newsroom/press-releases/2022-06-14-gartner-forecasts-worldwide-it-spending-to-grow-3-percent-in-2022 (дата звернення: 23.08.2022).
52. Gartner: Global Research and Advisory Company // Gartner [Електронний ресурс]. URL: https://www.gartner.com/en (дата звернення: 21.08.2021).
53. Gartner Says Worldwide IT Spending is Forecast to Grow 0.6 Percent in 2016 // Gartner [Електронний ресурс]. URL: https://www.gartner.com/en/newsroom/press-releases/2016-01-18-gartner-says-worldwide-it-spending-is-forecast-to-grow-1-percent-in-2016 (дата звернення: 20.08.2022).
54. Giarratano J. C., Riley G. D. Expert Systems: Principles and Programming / 4-е вид., Boston: Course Technology, 2004. 288 p.

55. Global Innovation Index 2020: Who Will Finance Innovation? // WIPO [Електронний ресурс]. URL: https://www.wipo.int/edocs/pubdocs/en/wipo_pub_gii_2020.pdf (дата звернення: 21.08.2021).
56. Gray A. R., MacDonell S. G. Fuzzy logic for software metric models throughout the development life-cycle // Annual Conference of the North American Fuzzy Information Processing Society - NAFIPS. 1999. P. 258–262.
57. Hughes R. T. Expert judgement as an estimating method // Information and Software Technology. 1996. № 2 (38). P. 67–75.
58. ILO Brief: The impact of the Ukraine crisis on the world of work: Initial assessments // International Labor Organisation [Електронний ресурс]. URL: https://www.ilo.org/wcmsp5/groups/public/---europe/---ro-geneva/documents/briefingnote/wcms_844295.pdf (дата звернення: 20.08.2022).
59. Information Technology for Knowledge Management под ред. U. M. Borghoff, R. Pareschi, Berlin: Springer Berlin, 1998.
60. IT Industry Outlook 2021 // CompTIA [Електронний ресурс]. URL: https://www.comptia.org/content/research/it-industry-trends-analysis (дата звернення: 21.08.2021).
61. IT Ukraine Report 2021 [Електронний ресурс]. URL: https://drive.google.com/file/d/1rDOzj3_hKgXfIj8czwIVzP8Ct4lBY5eW/view (дата звернення: 21.08.2021).
62. IT Outsourcing Industry - Statistics & Facts // Statista [Електронний ресурс]. URL:https://www.statista.com/topics/1900/it-outsourcing/#topicHeader__wrapper (дата звернення: 24.08.2022).
63. Jacobson D. Corporate Tax and Industrial Policy in Ireland: Selections from Upsetting the Applecart. Dublin, 2018.
64. Keynes J. M. The General Theory of Employment // The Quarterly Journal of Economics. 1937. № 2 (51). P. 209–223.
65. Know your customer // PWC [Електронний ресурс]. URL: https://www.pwc.com/gx/en/financial-services/assets/pwc-kyc-anti-money-laundering-guide-2013.pdf (дата звернення: 24.08.2021).

66. Knowledge-based systems, Four-Volume Set. Techniques and applications под ред. C. T. Leondes, Cambridge: Academic Press, 2000. 1449 p.
67. The Largest Technology Companies In 2019: Apple Reigns As Smartphones Slip And Cloud Services Thrive // Forbes [Електронний ресурс]. URL: https://www.forbes.com/sites/jonathanponciano/2019/05/15/worlds-largest-tech-companies-2019/?sh=3f75291e734f (дата звернення: 21.08.2021).
68. Lee C. C. Fuzzy Logic in Control Systems: Fuzzy Logic Controller—Part I // IEEE Transactions on Systems, Man and Cybernetics. 1990. № 2 (20). P. 404–418.
69. Lehman T. J., Sharma A. Software Development as a Service: Agile Experiences 2011. P. 749–758.
70. Machacha L. L., Bhattacharya P. A fuzzy-logic-based approach to project selection // IEEE Transactions on Engineering Management. 2000. № 1 (47). P. 65–73.
71. McConnell S. Software Estimation: Demystifying the Black Art. Redmond, WA, USA: Microsoft Press, 2006.
72. McCoy F. A. Measuring success: Establishing and maintaining a baseline // PMI Annual Seminar & Symposium, Montreal. 1986.
73. Might R. J., Fischer W. A. The role of structural factors in determining project management success // IEEE Transactions on Engineering Management. 1985. № 2 (EM-32). P. 71–77.
74. Mirza M. N., Pourzolfaghar Z., Shahnazari M. Significance of Scope in Project Success // Procedia Technology. 2013. (9). P. 722–729.
75. Miyazaki Y. [та ін.]. Robust regression for developing software estimation models // Journal of Systems and Software. 1994. № 1 (27). P. 3–16.
76. Morris P. W. G., Hough G. H. The anatomy of major projects: A study of the reality of project management. Oxford: John Wiley and Sons, 1987.
77. Mueller M., Grindal K. Is It “Trade?” Data Flows and the Digital Economy // TPRC 46: The 46th Research Conference on Communication, Information and Internet Policy 2018. 2018.

78. Nassif L. N., Filho J. C. S., Nogueira J. M. Project Portfolio Selection in Public Administration Using Fuzzy Logic // Procedia - Social and Behavioral Sciences. 2013. (74). P. 41–50.
79. Neely A., Gregory M., Platts K. Performance measurement system design: a literature review and research agenda // International journal of operations & production management. 1995. № 4 (15). P. 80–116.
80. Number of ICT professionals worldwide 2019-2023 // Statista [Електронний ресурс]. URL: https://www.statista.com/statistics/1126677/it-employment-worldwide/ (дата звернення: 21.08.2021).
81. Peker M. Expert system-based educational tool for practice lessons in a vocational high school // Energy Education Science and Technology Part B: Social and Educational Studies. 2013. (4). P. 2165–2176.
82. Performance appraisal tools // Management Study Guide [Електронний ресурс]. URL: https://www.managementstudyguide.com/performance-appraisal-tools.htm (дата звернення: 24.08.2021).
83. Pfleeger S. L., Atlee J. M. Software Engineering: Theory and Practice. 4th-е вид., USA: Pearson, 2010.
84. Phil M. Comparative Analysis of Different Agile Methodologies // International Journal of Computer Science and Information Technology Research. 2015. № 1 (3).
85. Pinto J. K., Slevin D. P. Critical factors in successful project implementation // IEEE Transactions on Engineering Management. 1987. № 1 (EM-34). P. 22–27.
86. Polak J., Wójcik P. Knowledge Management in IT Outsourcing/Offshoring Projects // PM World Journal. 2015. № VIII (IV).
87. PwC | Global Top 100 companies by market capitalisation 31 March 2018 update // PwC [Електронний ресурс]. URL: https://www.pwc.com/gx/en/audit-services/assets/pdf/global-top-100-companies-2018-report.pdf (дата звернення: 03.07.2021).
88. PwC | Global Top 100 companies-by market capitalisation May 2022 // PwC [Електронний ресурс]. URL: https://www.pwc.com/gx/en/audit-

services/publications/top100/pwc-global-top-100-companies-by-market-capitalisation-2022.pdf (дата звернення: 20.08.2022).

89. Rad P. F., Cioffi D. F. Work and resource breakdown structures for formalized bottom-up estimating // Cost engineering. 2000. № 2 (46). P. 31–37.
90. Rosenau M. D., Githens G. D. Successful project management: a step-by-step approach with practical examples / 4-е вид., United States: Wiley, 2005.
91. SAFe® for Lean Enterprises // Scaled Agile, Inc. [Електронний ресурс]. URL: https://www.scaledagileframework.com/ (дата звернення: 22.08.2021).
92. Schlegel D. R., Shapiro S. C. Concurrent reasoning with inference graphs Springer Verlag, 2014. P. 138–164.
93. Shapiro S. C., Rapaport W. J. The SNePS family // Computers and Mathematics with Applications. 1992. № 2–5 (23). P. 243–275.
94. Sharma S., Sarkar D., Gupta D. Agile Processes and Methodologies: A Conceptual Study // International Journal on Computer Science and Engineering. 2012. (4).
95. Shortliffe E. H., Kaiser H. J. Medical Expert Systems—Knowledge Tools for Physicians // Western Journal of Medicine. 1986. № 6 (145). P. 830.
96. Shrnhur A. J., Levy O., Dvir D. Mapping the dimensions of project success // Project management journal. 1997. № 2 (28). P. 5–13.
97. Shumanskaya M. Improvement of Outsourcing Information Technology Services from Other Countries to Finland. 2011. 1–49 p.
98. Slegten K. Dynamic Systems Development Method (DSDM) [Електронний ресурс]. URL: http://www.students.science.uu.nl/~slegt001/me/final_5767202_Slegten.pdf (дата звернення: 22.08.2021).
99. Software Development Outsourcing to Ukraine: 9 Reasons Why It’s the Best Choice // Aimprosoft [Електронний ресурс]. URL: https://www.aimprosoft.com/blog/it-outsourcing-ukraine (дата звернення: 21.08.2021).

100. Sokolovska Z., Dudnyk O. Devising a technology for managing outsourcing IT-projects with the application of fuzzy logic // Eastern-European Journal Of Enterprise Technologies. 2021. № 2(3 (110). P. 52-65.

101. Stage-Gate Model // Stage-Gate International [Електронний ресурс]. URL: https://www.stage-gate.com/stage-gate-model (дата звернення: 22.08.2021).

102. The Stage-Gate® Model: An Overview // Stage-Gate International [Електронний ресурс]. URL: https://www.stage-gate.com/wp-content/uploads/2018/06/wp10english.pdf (дата звернення: 22.08.2021).

103. Stewart J. MNE tax strategies and Ireland // Critical Perspectives on International Business. 2018. № 4 (14). P. 338–361.

104. Tech Ecosystem Guide to Ukraine // UNIT city [Електронний ресурс]. URL: https://data.unit.city/tech-guide/Tech_Ecosystem_Guide_To_Ukraine_Ua.pdf (дата звернення: 21.08.2021).

105. TopCoder Statistics: country rating // TopCoder [Електронний ресурс]. URL: https://community.topcoder.com/stat?c=country_avg_rating (дата звернення: 21.08.2021).

106. Trikha R. Which Country Would Win in the Programming Olympics? // HackerRank [Електронний ресурс]. URL: https://blog.hackerrank.com/which-country-would-win-in-the-programming-olympics (дата звернення: 21.08.2021).

107. Turner J. R., Speiser A. Programme management and its information systems requirements // International Journal of Project Management. 1992. № 4 (10). P. 196–206.

108. Turner J. R. The handbook of Project-Based Management: improving the processes for achieving strategic objectives. New York: McGraw-Hill, 1999.

109. Ukraine Named Offshoring Destination of the Year // Lviv IT cluster [Електронний ресурс]. URL: https://itcluster.lviv.ua/en/ukraine-offshoring-destination-of-the-year (дата звернення: 21.08.2021).

110. Ukraine: The country that codes. IT Industry in Ukraine. 2019 Market Report // N-ix [Електронний ресурс]. URL: https://s3-eu-west-1.amazonaws.com/new.n-

ix.com/uploads/2019/09/26/Software_development_in_Ukraine_2019_2020_IT_industry_market_report.pdf (дата звернення: 21.08.2021).

111. UNCTAD | Digital Economy Report 2019 // UNCTAD [Електронний ресурс]. URL: https://unctad.org/system/files/official-document/der2019_en.pdf (дата звернення: 21.08.2021).
112. UNCTAD | Digital Economy Report 2021 // UNCTAD [Електронний ресурс]. URL: https://unctad.org/system/files/official-document/der2021_en.pdf (дата звернення: 20.08.2022).
113. UNCTAD | Handbook of Statistics // UNCTAD [Електронний ресурс]. URL: https://unctad.org/system/files/official-document/tdstat46_en.pdf (дата звернення: 20.08.2022).
114. UNCTADstat: World Statistical Database // UNCTAD [Електронний ресурс]. URL: https://unctadstat.unctad.org/EN/Index.html (дата звернення: 21.08.2021).
115. Vasyuk K. IT is Ukraine's fastest-growing sector. When the war is over, it will be the driver of our economy once again. - DIGITALEUROPE // DigitalEurope [Електронний ресурс]. URL: https://www.digitaleurope.org/news/it-is-ukraines-fastest-growing-sector-when-the-war-is-over-it-will-be-the-driver-of-our-economy-once-again/ (дата звернення: 20.08.2022).
116. Vedsmand T., Cooper R., Kielgast S. Integrating Agile with Stage-Gate® – How New Agile-Scrum Methods Lead to Faster and Better Innovation // Innovation Management. SE. 2016. P. 1–15.
117. Węglarz A., Gilewski P. G. Application of expert systems in the construction sector // MATEC Web of Conferences. 2017. (117). P. 176.
118. What's the Difference? Agile vs Scrum vs Waterfall vs Kanban // Smartsheet [Електронний ресурс]. URL: https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban (дата звернення: 22.08.2021).
119. Where to find the best developers in 2019? Report based on +550 technical assesments // SkillValue [Електронний ресурс]. URL: https://digital-platform.pentalog.com/rs/232-EET-259/images/Rapport-IT-EN-2019.pdf (дата звернення: 21.08.2021).

120. World Economic Situation And Prospects: February 2021 Briefing, No. 146 | Department of Economic and Social Affairs // United Nations DESA [Електронний ресурс]. URL: https://www.un.org/development/desa/dpad/publication/world-economic-situation-and-prospects-february-2021-briefing-no-146/ (дата звернення: 03.07.2021).
121. World Trade Organization Stats Dashboard // WTO [Електронний ресурс]. URL: https://stats.wto.org/ (дата звернення: 20.08.2022).
122. World Trade Statistical Review 2021 // WTO [Електронний ресурс]. URL: https://www.wto.org/english/res_e/statis_e/wts2021_e/wts2021_e.pdf (дата звернення: 23.08.2022).
123. Yarova M. 21 companies from Ukraine among the world's top 100 outsourcing service providers // AIN.UA [Електронний ресурс]. URL: https://ain.ua/en/2020/02/19/20-companies-from-ukraine-among-global-outsourcing-100 (дата звернення: 21.08.2021).
124. Zadeh L. A. Fuzzy sets // Information and Control. 1965. № 3 (8). P. 338–353.
125. Zadeh L. A. The concept of a linguistic variable and its application to approximate reasoning—I // Information Sciences. 1975. № 3 (8). P. 199–249.
126. Zadeh L. A. PRUF—a meaning representation language for natural languages // International Journal of Man-Machine Studies. 1978. № 4 (10). P. 395–460.

ДОДАТОК А

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Dudnyk, O., Sokolovska, Z. (2022). Application of Fuzzy Expert Systems in IT Project Management. In M. Mircea, & T. M. M. Nguyen (Eds.), Project Management - New Trends and Applications. IntechOpen. https://doi.org/10.5772/intechopen.102439 *(0,62 д.а., особистий внесок здобувача: розроблено алгоритм оцінки виробничого потенціалу IT-компанії – 0,4 д.а.)*.
2. Дудник Олексій, Соколовська Зоя, "Devising a technology for managing outsourcing IT-projects with the application of fuzzy logic" // Eastern-European Journal Of Enterprise Technologies (Scopus), DOI: 10.15587/1729-4061.2021.224529, https://doi.org/10.15587/1729-4061.2021.224529, Indexing: Scopus, CrossRef, American Chemical Society, EBSCO, Index Copernicus, MIAR, CNKI, Ulrich's Periodical Directory, Neliti, 1findr, BASE, WorldCat, Polska Bibliografia Naukowa (*0,58 д.а., особистий внесок здобувача: розроблено модифікацію Stage-Gate фреймворку на основі нечіткої логіки – 0,45 д.а.*).
3. Дудник Олексій, Зоя Соколовська, "Forecasting development trends in the information technology industry using fuzzy logic" // Eastern-European Journal of Enterprise Technologies. 1, 74–85 (2023). (Scopus) https://doi.org/10.15587/1729-4061.2023.267906, Indexing: Scopus, CrossRef, American Chemical Society, EBSCO, Index Copernicus, MIAR, CNKI, Ulrich's Periodical Directory, Neliti, 1findr, BASE, WorldCat, Polska Bibliografia Naukowa (*0,5 д.а., особистий внесок здобувача: розроблено алгоритм прогнозування стану IT-індустрії – 0,35 д.а.*).
4. Дудник Олексій, Соколовська Зоя, "Modern Technologies for Management of Outsourcing IT Projects" // Business Inform (фаховий журнал), DOI: 10.32983/2222-4459-2021-2-62-72, https://doi.org/10.32983/2222-4459-2021-2-62-72 (*0,42 д.а., особистий внесок здобувача: проведено аналіз проблем та сучасних підходів до управління IT-проєктами – 0,3 д.а.*).

Наукові праці, які засвідчують апробацію матеріалів дисертації

5. Дудник Олексій, “Удосконалення процесів управління портфелем проєктів в IT-компаніях” // Економічна кібернетика: теорія, практика та напрямки розвитку (тези), https://economics.opu.ua/files/science/ek_kiber/2017/52.pdf *(0,12 д.а.).*
6. Дудник Олексій, “Аналіз оптимальної структури бази знань для нечітких експертних систем” // XI Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем” (тези, онлайн конференція - 11-12 квітня 2019 р), https://mpsesm.org/index.php/mpsesm/mpsesm-xi/paper/view/786/680 *(0,16 д.а.).*
7. Дудник Олексій, “Процес вибору виконавців IT-проєктів у портфоліо проєктів” // Економічна кібернетика: теорія, практика та напрямки розвитку” (тези), https://economics.opu.ua/files/science/ek_kiber/2018/50.pdf *(0,12 д.а.).*
8. Дудник Олексій, “Аналіз процесів управління портфелем проєктів в IT-компаніях” // X Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем” (тези, онлайн конференція - 5-6 квітня 2018 р), https://mpsesm.org/index.php/mpsesm/mpsesm-x/paper/view/670/571 *(0,16 д.а.).*
9. Дудник Олексій, “Переваги використання нечіткої логіки в процесі управління IT-проєктами” // XIII Міжнародна науково-практична Інтернет-конференція “Сучасні проблеми моделювання соціально-економічних систем” (тези, онлайн конференція - 8-9 квітня 2021 р), https://mpsesm.org/book/2021/thesis03-910.html *(0,16 д.а.).*
10. Дудник Олексій, “Економіка та апарат нечітких експертних систем: тренди розвитку та застосування” // Економічна кібернетика: теорія, практика та напрямки розвитку” (тези), https://economics.net.ua/wp-content/uploads/2023/01/tezy.pdf *(0,2 д.а.).*

Наукові праці, які додатково відображають наукові результати дисертації

11. Соколовська З.М., Дудник О.О. Експертні системи: теорія та прикладне застосування: [монографія]. Одеса: Екологія, 2022. 408 с. *(23,72 д.а., особистий внесок здобувача: розділи 3, 6, 7 − 10 д.а.).*

ДОДАТОК Б

Відомості про апробацію результатів дисертації



INVERS

Invers B.V.

01.09.2022

"Validating"
CTO Invers B.V.
Frank Nolden

INTEGRATION ACT

This act is a confirmation of the performed analysis on the potential research usage of Oleksii Dudnyk, the graduate student of the "Odessa Polytechnic" National University, within company's subject area. The analysis conducted by Invers B.V. from 01.06.2022 to 31.08.2022 shows that Oleksii's doctoral thesis on "Innovative technologies in IT companies project portfolio management" concerns the usage of fuzzy logic rule based systems, which could be used in company's practice as follows:

1. For already existing rule based system that is partially used for the purpose of bank transactions categorization, the usage of the proposed SNEPS fuzzy inference core could potentially reduce the dependance on crisp values and would allow to develop a bigger set of rules operating under the same time constraints.
2. SNEPS fuzzy inference core can be used as a backup option in case Machine Learning models for bank transactions categorization failed to provide a result.

Invers B.V.
CTO Frank Nolden

INVERS

Informatie Technologie Systemen

01.09.2022

"Затверджую"
Технічний директор Invers B.V.
Frank Nolden

_____

АКТ ВПРОВАДЖЕННЯ
наукових результатів дисертаційної роботи Дудника О. О. на тему
"Інноваційні технології в управлінні портфелем проектів IT-компаній"

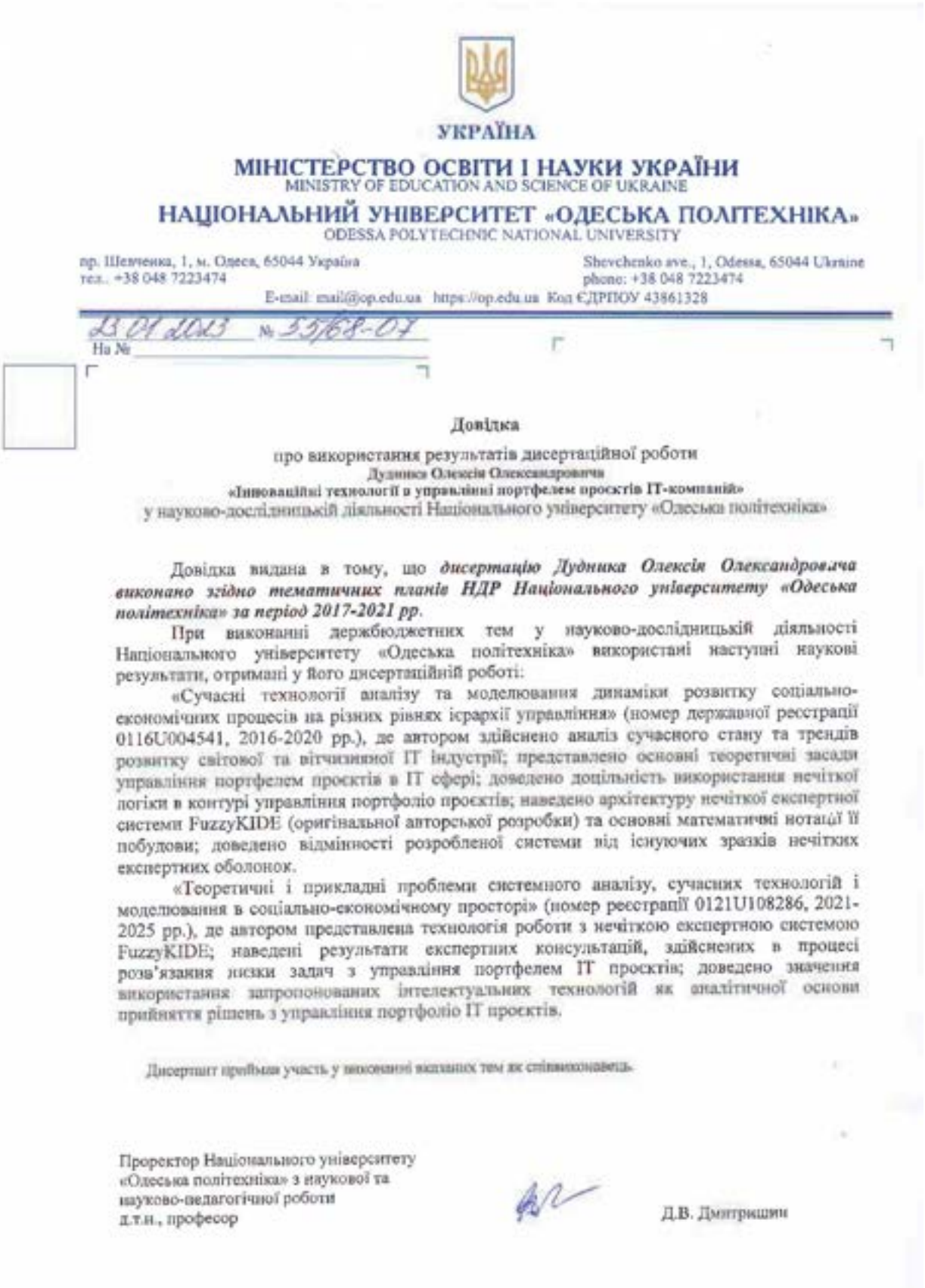
Цей акт є підтвердженням проведеного аналізу можливостей використання наукових досліджень аспіранта Національного університету «Одеська політехніка» Олексія Дудника в рамках діяльності компанії. Аналіз, проведений компанією Invers B.V. з 01.06.2022 по 31.08.2022, показує, що докторська дисертація Олексія на тему «Інноваційні технології в управлінні портфелем проектів IT-компаній» стосується використання систем на основі правил нечіткої логіки, які можуть бути використані в практиці компанії наступним чином:

1. Для вже існуючої системи на основі правил, яка частково використовується для категоризації банківських транзакцій, використання запропонованого ядра нечіткого логічного виводу SNEPS зменшить залежність від чітких значень і дозволить розробити більший набір правил, вивід по яким буде вкладатися у вже існуючі часові обмеження.
2. Ядро нечіткого виводу SNEPS може використовуватися як резервний варіант, якщо моделі машинного навчання для категоризації банківських транзакцій не дали результату.

Від Invers B.V.
Технічний директор Frank Nolden

ДОДАТОК В

Довідка про використання результатів дисертаційної роботи у науково-дослідницької діяльності



УКРАЇНА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
ODESSA POLYTECHNIC NATIONAL UNIVERSITY

пр. Шевченка, 1, м. Одеса, 65044 Україна
тел.: +38 048 7223474

Shevchenko ave., 1, Odessa, 65044 Ukraine
phone: +38 048 7223474

E-mail: mail@op.edu.ua https://op.edu.ua Код ЄДРПОУ 43861328

23.01.2023 № 3568-07
На №

Довідка

про використання результатів дисертаційної роботи
Дудника Олексія Олександровича
«Інноваційні технології в управлінні портфелем проєктів IT-компаній»
у науково-дослідницькій діяльності Національного університету «Одеська політехніка»

Довідка видана в тому, що ***дисертацію Дудника Олексія Олександровича виконано згідно тематичних планів НДР Національного університету «Одеська політехніка» за період 2017-2021 рр.***

При виконанні держбюджетних тем у науково-дослідницькій діяльності Національного університету «Одеська політехніка» використані наступні наукові результати, отримані у його дисертаційній роботі:

«Сучасні технології аналізу та моделювання динаміки розвитку соціально-економічних процесів на різних рівнях ієрархії управління» (номер державної реєстрації 0116U004541, 2016-2020 рр.), де автором здійснено аналіз сучасного стану та трендів розвитку світової та вітчизняної IT індустрії; представлено основні теоретичні засади управління портфелем проєктів в IT сфері; доведено доцільність використання нечіткої логіки в контурі управління портфоліо проєктів; наведено архітектуру нечіткої експертної системи FuzzyKIDE (оригінальної авторської розробки) та основні математичні нотації її побудови; доведено відмінності розробленої системи від існуючих зразків нечітких експертних оболонок.

«Теоретичні і прикладні проблеми системного аналізу, сучасних технологій і моделювання в соціально-економічному просторі» (номер реєстрації 0121U108286, 2021-2025 рр.), де автором представлена технологія роботи з нечіткою експертною системою FuzzyKIDE; наведені результати експертних консультацій, здійснених в процесі розв'язання низки задач з управління портфелем IT проєктів; доведено значення використання запропонованих інтелектуальних технологій як аналітичної основи прийняття рішень з управління портфоліо IT проєктів.

Дисертант приймав участь у виконанні вказаних тем як співвиконавець.

Проректор Національного університету
«Одеська політехніка» з наукової та
науково-педагогічної роботи
д.т.н., професор Д.В. Дмитришин

ДОДАТОК Г

Довідка про впровадження результатів дисертаційної роботи у навчальний процес

УКРАЇНА

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
ODESSA POLYTECHNIC NATIONAL UNIVERSITY

пр. Шевченка, 1, м. Одеса, 65044 Україна
тел.: +38 048 7223474
Shevchenko ave., 1, Odessa, 65044 Ukraine
phone: +38 048 7223474
E-mail: mail@op.edu.ua https://op.edu.ua Код ЄДРПОУ 43861328

23.01.2023 № 34/68-07
На №

Довідка

про впровадження результатів дисертаційної роботи
Дудника Олексія Олександровича
«Інноваційні технології в управлінні портфелем проектів IT-компаній»
у навчальний процес Національного університету «Одеська політехніка»

Довідка видана у тому, що в програмах, навчально-методичних матеріалах та курсах лекцій з дисциплін «Управління проектами інформатизації», та «Експертні системи в формуванні стратегії розвитку підприємства», що вивчаються бакалаврами спеціальності 051 в межах освітньої програми «Економічна кібернетика» у 7 семестрі, використовуються наступні наукові результати, отримані у дисертації **Дудника Олексія Олександровича**:

- дисципліна «Управління проектами інформатизації», змістовний модуль «Інформаційні технології управління проектами», тема «Інформаційні та програмні засоби управління проектами»: аналіз та класифікація проблем управління IT проектами; визначення основних аспектів та відмінностей в управлінні портфоліо проектів; визначення особливостей управління проектами в IT аутсорсингових фірмах; порівняльний аналіз сучасних методологій управління IT-проектами; обґрунтування вибору технологічної бази управління IT проектами в умовах неточної інформації; аналіз програмних платформ підтримки проектних рішень; обґрунтування доцільності використання апарату нечіткої логіки в процесі управління проектами в умовах невизначеності.
- дисципліна «Управління проектами інформатизації», змістовний модуль «Процеси і підсистеми управління проектами», теми «Управління ресурсами та вартістю проекту», «Управління якістю та ризиками проекту», «Управління персоналом у проектах»: формування концепції та технології застосування Agile Stage-Gate фреймворку з використанням нечіткої логіки; технологія прийняття рішень з управління портфоліо проектів на програмних платформах нечітких експертних систем; приклади розв'язання основних проектних задач із застосуванням апарату нечіткої логіки.
- дисципліна «Експертні системи в формуванні стратегії розвитку підприємства»: проведення лабораторних занять із застосуванням програмної платформи нечіткої експертної системи FuzzyKIDE (оригінальна авторська розробка).

Використані результати дисертації Дудника Олексія Олександровича свідчать про глибоке опрацювання ним науково-методичних та прикладних засад застосування інноваційних технологій (інтелектуальних інформаційних технологій – нечітких експертних систем) в управлінні портфелем проектів IT компаній. Висновки та пропозиції відзначаються науковою новизною, можливістю реалізації у практичній діяльності важливих суб'єктів інформаційної економіки – IT підприємств, що істотно поліпшує матеріал означених дисциплін та сприяє підвищенню якості підготовки фахівців з економічної кібернетики.

Перший проректор, проректор з науково-педагогічної
та виховної роботи
д.т.н., професор С.А. Нестеренко