Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Бізнес-аналіз та моделювання процесів прийняття рішень в управлінні проектами
(тема)

Виконав:
здобувач _2_ року навчання, групи _САУм-23-2
Отрощенко А.О.
(прізвище, ініціали)
Спеціальність
124 Системний аналіз
(код і повна назва спеціальності)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Системний аналіз і управління
(повна назва освітньої програми)
Керівник доц. Ламтюгова С.М.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ПМ _____ Сидоров М.В.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту
Кафедра _прикладної математики
Рівень вищої освіти _другий (магістерський)
Спеціальність _124 Системний аналіз
(код і повна назва)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма _Системний аналіз і управління
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри ПМ _____
(підпис)
“ _25_ ” _листопада_ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві _Отрощенку Андрію Олександровичу
(прізвище, ім’я, по батькові)

1. Тема роботи _Бізнес-аналіз та моделювання процесів прийняття рішень в управлінні проєктами
затверджена наказом по університету від 22 листопада 2024 р. № 1228 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 6 січня 2025 р.

3. Вихідні дані до роботи вимоги клієнта до рішення бізнес задачі, бюджет на розробку ПЗ, прогноз на потенційний прибуток, перелік ризиків на аналіз ринку

4. Перелік питань, що потрібно опрацювати в роботі
1. Системний аналіз предметної області
2. Вибір і обґрунтування методу розв’язання
3. Процес прийняття рішень впродовж життєвого циклу розробки програмного забезпечення
4. Реалізація системи
5. Аналіз можливих застосувань

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

1. Схеми дерева рішень
2. Таблиці з даними для розрахунку
3. Таблиці рішень
4. Діаграми вимог
5. Приклади функціонування створеного шаблону

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Підбір та вивчення технічної літератури за темою роботи	25 листопада – 1 грудня 2024 р.	виконано
2	Вибір та обґрунтування методу	2 – 8 грудня 2024 р.	виконано
3	Розробка алгоритму і програми	9 – 22 грудня 2023 р.	виконано
4	Проведення аналітичних досліджень та розрахунків	23 – 29 грудня 2024 р.	виконано
5	Робота над текстом пояснювальної записки	30 грудня 2024 р. – 9 січня 2025 р.	виконано
6	Представлення роботи на рецензію в ЕК	10 січня 2025 р.	виконано

Дата видачі завдання _25_ листопада 2024 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Ламтюгова С.М.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 81 с., 4 табл., 20 рис., 1 дод., 17 джерел.

БІЗНЕС АНАЛІЗ, МОДЕЛЮВАННЯ ПРОЦЕСУ ПРИЙНЯТТЯ РІШЕНЬ, АУТСОРС, МЕНЕДЖМЕНТ ПРОЄКТІВ, ДЕРЕВО ПРИЙНЯТТЯ РІШЕНЬ, ЦІННІСТЬ РІШЕННЯ

Об’єкт дослідження – процеси прийняття рішень в управлінні IT-проєктами.

Мета роботи – розробка автоматизованої системи прийняття рішень для декількох проєктів з можливістю її масштабування на рівні компанії.

Методи дослідження – метод виявлення слабких і сильних сторін для низки найпопулярніших технік таких, як BPMN Моделювання, Моделювання рішень та SWOT-аналіз.

Кваліфікаційна робота присвячена розробці комплексної методології підтримки прийняття рішень у бізнес-аналізі для IT-проєктів, зокрема в контексті аутсорсингових компаній. У роботі розглядаються ключові аспекти життєвого циклу розробки програмного забезпечення, визначаються основні точки прийняття рішень та пропонуються ефективні методи автоматизації цього процесу. Особливу увагу приділено фазі pre-sale/discovery, на якій бізнес-аналітики виконують збір вимог, аналіз ринку та оцінку проєктів. Запропонований підхід використовує техніки моделювання рішень, зокрема дерева рішень, для оптимізації та структурування процесів прийняття рішень. Ця методологія забезпечує прозорість, масштабованість та ефективність аналізу доцільності проєктів і стратегічного планування.

Було проведено детальний аналіз інструментів для побудови дерев рішень, таких як Microsoft Excel, Lucidchart, Python (sklearn) та Draw.io. Кожен інструмент оцінено за його перевагами та недоліками, а також надано рекомендації залежно від складності завдань і доступних ресурсів. У роботі також розг-

лянуто реальні кейси застосування дерев рішень для різних сценаріїв: розробки повного продукту, інтеграції з існуючими системами та використання готових сервісів. Ці приклади дозволили оцінити економічну доцільність та ризики для кожного варіанту, а також забезпечити автоматизований розрахунок очікуваної вартості.

У роботі запропоновано шаблони для автоматизації ухвалення рішень на етапі discovery. Вони значно скорочують час і зусилля, необхідні для оцінки проєктів, та сприяють оптимізації процесу pre-sale. Результати дослідження підтвердили, що використання формалізованих підходів, таких як Decision Tree, суттєво знижує рівень невизначеності та покращує комунікацію між стейкхолдерами. Запропонована методологія та інструменти є адаптивними і можуть бути застосовані в широкому колі IT-компаній, які працюють у сфері аутсорсу. Фокус на автоматизації та моделюванні рішень забезпечує міцну основу для вдосконалення процесів прийняття рішень, підвищення успішності проєктів та побудови довгострокових відносин із клієнтами. Отримані результати сприяють розвитку галузі бізнес-аналізу, пропонуючи практичні рішення для підтримки прийняття рішень та їх автоматизації в управлінні IT-проєктами, вирішуючи як стратегічні, так і операційні завдання.

ABSTRACT

Introductory note: 81 pages, 4 tables, 20 figures, 1 appendix, 17 sources.

BUSINESS ANALYSIS, DECISION MODELING, OUTSOURCING, PROJECT MANAGEMENT, DECISION TREE, EXPECTED VALUE

The object of the study is decision-making processes in IT project management.

The purpose of the work is to develop an automated decision-making system for multiple projects with the potential for scaling at the company level.

Methods of research are the methods of identifying strengths and weaknesses for a range of the most popular techniques, such as BPMN Modeling, Decision Modeling, and SWOT Analysis.

The qualifying work focuses on developing a comprehensive methodology for decision-making support in business analysis for IT projects, particularly in the context of outsourcing companies. The study addresses critical aspects of the software development lifecycle, identifying key decision points and proposing effective methods for automating the decision-making process. The research highlights the importance of the pre-sale/discovery phase, where business analysts perform requirement gathering, market analysis, and project evaluation. The proposed approach leverages Decision Modeling techniques, specifically Decision Trees, to streamline and structure decision-making processes. This methodology ensures transparency, scalability, and efficiency in analyzing project feasibility and strategic planning.

An in-depth analysis of tools for constructing decision trees, such as Microsoft Excel, Lucidchart, Python (sklearn), and Draw.io, was conducted. Each tool was evaluated for its strengths and limitations, with recommendations provided based on task complexity and available resources. Real-world cases were explored to demonstrate the application of decision trees in scenarios such as full-stack product development, integration with existing systems, and the use of pre-built solutions. These

examples enabled economic feasibility assessments and automated expected value calculations for various decision paths.

The paper further introduces templates for automating decision-making during the discovery phase. These templates facilitate rapid evaluation of project proposals and help optimize the pre-sale process. The results demonstrate that the use of structured techniques like Decision Trees significantly reduces uncertainty and enhances collaboration among stakeholders. The developed methodology and tools are designed to be adaptable and applicable to a wide range of IT outsourcing companies. A focus on automation and decision modeling provides a robust framework for improving decision-making processes, increasing project success rates, and fostering long-term client relationships. The findings contribute to the broader field of business analysis by offering practical solutions for decision-making support and automation in IT project management, addressing both strategic and operational challenges.

ЗМІСТ

ВСТУП

Актуальність теми. Завдання бізнес-аналізу та моделювання процесів прийняття рішень у сфері управління проєктами охоплюють аналіз вимог, оцінку ризиків, планування ресурсів та управління зацікавленими сторонами. Бізнес-аналітик повинен мати можливість швидко приймати ефективні рішення, які сприятимуть досягненню цілей проєкту в умовах постійних змін.

Однією з ключових проблем у процесі реалізації цих задач є власне прийняття рішень, яке повинно базуватися на об’єктивних даних та враховувати численні фактори, які впливають на успіх проєкту. У цій роботі розглядаються основні техніки, що дозволяють аналітику структурувати процеси прийняття рішень та зробити їх більш ефективними.

Створена ефективна система підтримки прийняття рішень для управління проєктами в аутсорсингових IT-компаніях має забезпечувати оптимізацію управлінських процесів через застосування технік моделювання рішень (Decision Modeling), які допомагають структуровано підійти до аналізу та вибору рішень. Зокрема, використання анотацій BPMN (Business Process Model and Notation) та UML (Unified Modeling Language) дозволяє візуалізувати процеси та зробити їх прозорими для всіх учасників проєкту.

Актуальність роботи зумовлена тим, що в більшості компаній процес роботи над проєктами в умовах аутсорсу з точки зору проєктного менеджера та бізнес-аналітика є дуже схожим [1]. Тому, щоб скоротити витрати ресурсів на роботу, що повторюється (або дуже схожа), на рівні компанії є запит на оптимізацію процесів прийняття рішень [2].

Мета і завдання кваліфікаційної роботи. Метою кваліфікаційної роботи є розробка автоматизованої системи прийняття рішень для декількох проєктів з можливістю її масштабування на рівні компанії.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести огляд і аналіз сучасних методів та технік, що використовуються спеціалістами під час прийняття рішень в різних проєктах;

– виконати порівняльний аналіз таких технік з урахуванням слабких і сильних сторін;

– виконати аналіз технік на можливість автоматизації та масштабування.

Об'єктом дослідження є процеси прийняття рішень в управлінні IT-проєктами.

Предметом дослідження є методи та інструменти бізнес-аналізу, зокрема техніки моделювання прийняття рішень на основі BABOK, а також їх застосування в управлінні проєктами.

Методи дослідження. У роботі використовується метод виявлення слабких і сильних сторін для низки найпопулярніших технік таких, як BPMN Моделювання, Моделювання рішень та SWOT-аналіз.

Публікації. Результати, отримані у роботі, було представлено на III Міжнародній молодіжній науково-практичній конференції англійською мовою «Навчання і викладання: у світі після війни» (м. Харків, 8 листопада 2024 року) [3].

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Системний аналіз задачі розробки ефективної системи підтримки прийняття рішень для управління проєктами в аутсорсових ІТ-компаніях

Бізнес-аналіз – це дисципліна, яка займається вивченням потреб бізнесу та визначенням рішень для бізнес-завдань [1]. У контексті управління проєктами, особливо для ІТ-аутсорсингових компаній, бізнес-аналіз виступає містком між зацікавленими сторонами і технічними командами, забезпечуючи відповідність цілей проєкту цілям бізнесу. Роль бізнес-аналітика включає розуміння вимог клієнта, оцінку процесів та оптимізацію робочих процесів для підвищення ефективності та результативності. Досліджуючи поточні процеси та визначаючи області для поліпшення, бізнес-аналітики допомагають компаніям приймати стратегічні рішення, що є критично важливим на швидко мінливому та конкурентному ринку ІТ-аутсорсингу [2].

ІТ-аутсорсингові компанії часто працюють над проєктами для клієнтів із різних галузей, кожен з яких має унікальні потреби та цілі. Ця різноманітність вимагає надійного підходу до управління вимогами, зменшення ризиків і забезпечення високоякісних результатів [2]. Бізнес-аналіз дозволяє аутсорсинговим компаніям систематично розуміти і документувати ці вимоги, надаючи чіткість для команд розробки та створюючи міцну основу для успішного виконання проєктів. Крім того, техніки бізнес-аналізу, такі як моделювання вимог, аналіз зацікавлених сторін та оптимізація процесів, підтримують ефективну комунікацію між компанією та її клієнтами, що призводить до більш цілеспрямованих та ефективних результатів проєктів [4].

У швидко мінливій галузі ІТ компанії повинні адаптуватися до нових технологій, змінних очікувань клієнтів та динаміки ринку. Бізнес-аналіз стає дедалі важливішим для ІТ-аутсорсингових компаній, які прагнуть залишатися

конкурентоспроможними. Він дозволяє цим компаніям залишатися гнучкими, адаптуючи процеси та стратегії до унікальних викликів кожного клієнтського проєкту [1]. Крім того, зі зростанням попиту на аутсорсинг ефективний бізнес-аналіз є необхідним для управління складними проєктами, забезпечення високих стандартів і збереження конкурентних переваг на світовому ринку IT.

Переваги бізнес-аналізу в управлінні проєктами:

– покращене прийняття рішень: бізнес-аналіз забезпечує інсайти, які базуються на даних, що підтримують більш обґрунтовані рішення і техніки, такі як SWOT-аналіз, моделювання прийняття рішень, оцінка ризиків та аналіз витрат і вигод, дозволяють компаніям оцінити різні підходи, зважити потенційні ризики та вибрати найбільш доцільний шлях [5];

– підвищення ефективності та якості: ідентифікуючи неефективності процесів та вузькі місця, бізнес-аналіз сприяє оптимізації операцій і це не лише заощаджує час та ресурси, але й підвищує якість результатів, оскільки проєкти краще відповідають потребам клієнтів [6];

– зменшення ризиків: бізнес-аналіз включає комплексну оцінку ризиків, що дозволяє компаніям проактивно ідентифікувати та усувати потенційні виклики і особливо це цінно в аутсорсингу, де важливо зберігати довіру клієнтів і надавати надійні результати;

– стратегічна узгодженість: бізнес-аналітики забезпечують, щоб кожен проєкт відповідав стратегічним цілям клієнта, сприяючи довготривалим партнерським відносинам і підвищуючи ймовірність повторного співробітництва, тож вона також підвищує задоволеність клієнтів, оскільки реалізовані рішення більшою мірою відповідають їхнім специфічним вимогам.

Бізнес-аналіз також допомагає підвищити ефективність внутрішньої взаємодії в командах, забезпечуючи прозорість і узгодженість на всіх етапах проєкту. Завдяки використанню таких методів, як управління вимогами, аналіз пріоритетів і створення дорожніх карт, компанії можуть зменшити кількість конфліктів між командами розробників, тестувальників і зацікавлених сторін. Це сприяє уникненню непорозумінь, скороченню часу на доопрацювання і

створенню продуктів, які краще відповідають очікуванням клієнтів. Таким чином, бізнес-аналіз стає ключовим інструментом для досягнення високої продуктивності та забезпечення довгострокового успіху проєктів.

1.2 Аналіз сценаріїв вирішення задачі розробки ефективної системи підтримки прийняття рішень для управління проєктами в аутсорсових IT-компаніях

Предметна область охоплює процеси бізнес-аналізу та прийняття рішень, що використовуються в управлінні проєктами для підвищення їх ефективності та забезпечення якісної оцінки ризиків і вигод. Управління проєктами передбачає застосування різних підходів, інструментів та моделей для прийняття стратегічних і тактичних рішень, таких як аналіз вимог, оцінка ризиків і ресурсів, а також управління зацікавленими сторонами.

Для аналізу бізнес-процесів та прийняття рішень у проєктах використовують методи, що включають наступні техніки:

– моделювання прийняття рішень (Decision Modeling (BABOK)) – техніка, що полягає у структуризації процесів прийняття рішень через створення моделі, яка описує правила та результати рішень та дозволяє візуалізувати і полегшити процеси прийняття рішень [1];

– моделювання бізнес-процесів (BPMN) – використання нотацій, таких як BPMN (Business Process Model and Notation) для опису та оптимізації процесів прийняття рішень [5];

– SWOT-аналіз – метод аналізу внутрішніх і зовнішніх факторів, які можуть вплинути на проєктні рішення [6];

– UML (Unified Modeling Language) – уніфікована мова моделювання, яка допомагає візуалізувати різні аспекти проєкту, включаючи прийняття рішень;

– аналіз ризиків – оцінка ризиків, пов'язаних з різними варіантами рішень, що дозволяє вибрати найменш ризиковані підходи [7].

Основні техніки та їх аналіз детально розглянемо далі. За результатами дослідження було прийнято рішення використати техніку моделювання прийняття рішень для створення автоматизованої системи, як ту, що найбільше підходить під вимоги аустосрсових компаній та специфіку роботи в них бізнес-аналітика.

1.3 Формальна та змістовна постановка задачі

У сучасних аутсорсових IT-компаніях процес управління проєктами містить багато повторюваних завдань та типових рішень, що може призводити до зайвих витрат ресурсів та підвищення ризиків. Багато компаній стикаються з проблемою оптимізації цих процесів, щоб підвищити ефективність роботи команд і мінімізувати ризики. Постійне повторення подібних дій в управлінні проєктами збільшує витрати часу та ресурсів, що на рівні компанії викликає запит на створення більш стандартизованих та автоматизованих підходів до прийняття рішень.

Задача полягає в розробці системи підтримки прийняття рішень для управління IT-проєктами, яка оптимізує повторювані процеси та зменшує ризики. Основою системи є застосування технік Decision Modeling, а також інструментів для візуалізації процесів, таких як BPMN, UML або Flowcharts. Система допоможе стандартизувати та формалізувати типові управлінські дії, створюючи темплейти для однотипних завдань. Це дозволить скоротити час на прийняття рішень, зробить процес більш керованим та відкриє можливості для часткової автоматизації. Розроблені темплейти повинні бути універсальними та адаптивними для застосування в різних проєктах.

Очікується, що розроблена система забезпечить оптимізацію процесів прийняття рішень, зменшить час на повторювані завдання, зробить процеси прозорішими та більш ефективними, а також сприятиме зниженню ризиків. Система має сприяти зменшенню людського фактора та ризиків, пов’язаних із не-

узгодженістю рішень на різних рівнях управління.

1.4 Постановка задач дослідження

Метою дослідження є розробка системи підтримки прийняття рішень для управління проєктами в IT-компаніях. У ході дослідження буде проведено аналіз популярних технік, які використовуються серед бізнес-аналітиків в IT-компаніях, таких як SWOT-аналіз, моделювання рішень, BPMN та UML-діаграми, з порівнянням їх сильних та слабких сторін [8].

Для досягнення поставленої мети необхідно виконати наступні завдання:

– проаналізувати існуючі методи та техніки прийняття рішень у проєктах;

– виконати порівняльний аналіз їх переваг та недоліків;

– оцінити можливість їх автоматизації та адаптації до потреб компанії;

– оцінити складність і ресурсозатратність підтримки такої системи;

– зробити висновки про вибрану техніку;

– розробити темплейт, згідно якого аналітик зможе приймати рішення в процесі управління проєктом.

2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ’ЯЗАННЯ

2.1 Опис техніки «Моделювання рішень» (Decision modeling)

Є певні стандартні бізнес-процеси, які кожна організація виконує щодня. Через їх частоту та важливість, ці бізнес-процеси повинні залишатися ефективними та результативними. Такі процеси ідеально підходять для вдосконалення, особливо, якщо ці вдосконалення можуть допомогти виявити та усунути вузькі місця.

Для покращення цих процесів може бути необхідно оптимізувати та автоматизувати деякі етапи. Наприклад, такі етапи, як затвердження, є ідеальними кандидатами для автоматизації, і модель прийняття рішень може бути використана для виявлення цих етапів. Моделі прийняття рішень – це візуальне представлення процесу, яке показує, як дані та знання об'єднуються для прийняття конкретного бізнес-рішення [1].

Модель пов'язана з бізнес-процесами, бізнес-правилами, ресурсами та показниками ефективності в організації. Наприклад, правило поведінки може бути таким: якщо адреса виставлення рахунку клієнта збігається з деталями кредитної картки, замовлення повинно бути оброблено [4].

Моделі прийняття рішень можуть використовуватися як для простих, так і для складних рішень.

Прості моделі використовують одну таблицю рішень або дерево рішень, щоб показати, як набір бізнес-правил працює на основі даних і формує рішення.

Складні рішення складаються з кількох простих рішень, об'єднаних в одне. Усі підходи до моделювання рішень включають три основні елементи [1]:

– рішення;

– інформація;

– знання.

Однак існують різні способи створення моделі рішень.

Наприклад, таблиця рішень використовується для відображення всіх біз-

нес-правил, необхідних для прийняття рішення.

2.1.1 Таблиці рішень (Decision table)

Бізнес-рішення приймаються на основі конкретного набору вхідних даних і за допомогою певних бізнес-правил для вибору одного з можливих результатів. Таблиця рішень – це простий і зручний спосіб представити ці правила у вигляді таблиці [1]. У ній кожен рядок (або стовпець) є окремим правилом, а кожен стовпець (або рядок) – умовою цього правила. Коли всі умови для певного правила відповідають наданим вхідним даним, вибирається дія чи результат, який відповідає цьому правилу.

У таблиці рішень зазвичай є кілька колонок з умовами, які відповідають конкретним елементам даних, а також кілька колонок з діями чи результатами. У кожному рядку зазначені умови, які перевіряються на відповідність вхідним даним. Якщо всі умови в рядку або залишаються порожніми, або є правильними, то це правило спрацьовує, і вибирається дія чи результат, зазначений у відповідній колонці. На рис. 2.1 наведений приклад використання такої техніки [1].

Eligibility Rules		
Loan Amount	**Age**	**Eligibility**
<=1000	>18	Eligible
	<=18	Ineligible
1000–2000	>21	Eligible
	<=21	Ineligible
>2000	>=25	Eligible
	<25	Ineligible

Рисунок 2.1 – Таблиця рішень

2.1.2 Дерева рішень (Decision tree)

Дерева рішень також використовуються для представлення набору бізнес-правил. Кожен шлях до листового вузла дерева рішень відповідає одному правилу. Кожен рівень дерева представляє певний елемент даних, а гілки, що йдуть далі, вказують на різні умови, які мають бути виконані, щоб продовжити рух по цій гілці. Дерева рішень можуть бути дуже ефективними для представлення певних наборів правил, особливо тих, що стосуються сегментації клієнтів.

Як і в таблицях рішень, дерево рішень вибирає одну з доступних дій або результатів (листовий вузол, що знаходиться внизу або праворуч) на основі конкретних значень, переданих елементами даних, які представляють вузли розгалуження.

У наведеному дереві рішень на рис. 2.2 правила дерева діляться спільними умовами (які представляються попередніми вузлами дерева) [1].

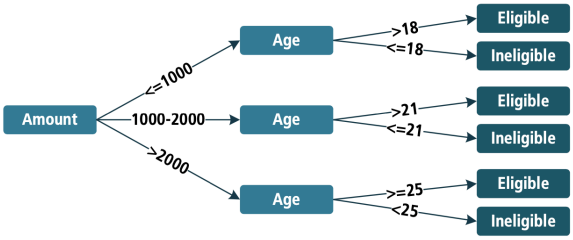

Рисунок 2.2 – Дерево рішень

2.1.3 Діаграми вимог до рішень (Decision Requirements Diagram)

Діаграма вимог до рішень – це візуальне представлення інформації, знань і процесу прийняття рішень, що використовується для складних бізнес-рішень. Діаграми вимог до рішень містять такі елементи:

– рішення: зображаються у вигляді прямокутників – кожне рішення приймає набір вхідних даних і обирає один із можливих результатів, застосо-

вуючи бізнес-правила та іншу логіку;

– вхідні дані: зображаються у вигляді овалів, що представляють дані, які повинні бути передані як вхідні для прийняття рішення на діаграмі;

– моделі бізнес-знань: показані у вигляді прямокутників із зрізаними кутами – вони представляють набори бізнес-правил, таблиці рішень, дерева рішень або навіть предиктивні аналітичні моделі, які точно описують, як слід приймати рішення;

– джерела знань: відображаються у вигляді документа, що представляє оригінальні джерела (документи або людей), на основі яких було отримано необхідну логіку прийняття рішень.

Ці елементи пов’язані між собою, утворюючи мережу, яка показує розбиття складного процесу прийняття рішень на простіші складові. Суцільні стрілки вказують на інформаційні вимоги для прийняття рішення. Такі вимоги можуть з'єднувати вхідні дані з рішенням, щоб показати, що для прийняття цього рішення потрібні певні дані, або можуть зв'язувати два рішення між собою.

Моделі бізнес-знань, які описують, як прийняти конкретне рішення, можуть бути пов’язані з цим рішенням за допомогою пунктирних стрілок для відображення вимог до знань. Джерела знань можуть бути пов'язані з рішенням за допомогою пунктирної стрілки із закругленими краями, що показує, що джерело знань (наприклад, документ або людина) є авторитетом для цього рішення. Це називається вимогою до авторитету.

На рис. 2.3 наведено приклад діаграми вимог [1].

2.1.4 Особливості до використання техніки моделювання прийняття рішень

Сильні сторони (плюси) техніки моделювання прийняття рішень:

– моделі рішень легко ділити з зацікавленими сторонами, що сприяє спільному розумінню і підтримує аналіз впливу;

– можна поєднувати кілька перспектив, особливо, якщо використовується діаграма;

– спрощує процес прийняття складних рішень, усуваючи управління бізнес-правилами з самого процесу;

– допомагає керувати великою кількістю правил у таблицях рішень, групуючи їх за рішеннями і це також сприяє повторному використанню правил;

– такі моделі підходять для автоматизації на основі правил, пошуку даних, предиктивної аналітики, а також для ручних рішень або проєктів бізнес-аналітики.

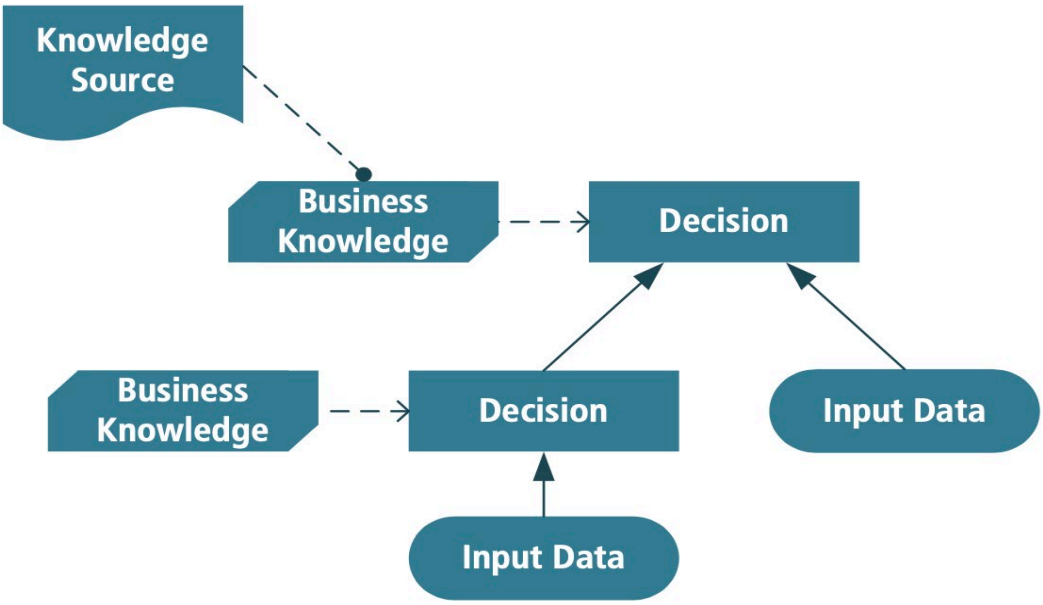

Рисунок 2.3 – Діаграма вимог до рішень

Слабкі сторони (мінуси) техніки моделювання прийняття рішень:

– додає ще один тип діаграми при моделюванні бізнес-процесів, що містять рішення і це може ускладнити процес, якщо рішення є простим і тісно пов'язаним із процесом;

– може обмежити правила лише тими, які потрібні для відомих рішень, і не дозволить фіксувати правила, що не стосуються поточного рішення;

– визначення моделей рішень може створити ілюзію, що організація має стандартизований підхід до прийняття рішень, хоча насправді це може бути не

так, а також це може "зафіксувати" організацію на існуючому підході до прийняття рішень;

– оскільки ці моделі можуть зачіпати міжорганізаційні межі, може виникнути складність у отриманні необхідного затвердження;

– бізнес-термінологія повинна бути чітко визначена, а спільні визначення розроблені, щоб уникнути проблем з якістю даних, які можуть вплинути на автоматизовані рішення, тому, відповідно, потрібні зусилля для підтримки такої документації з інформацією в актуальному стані.

2.2 Моделювання процесів в BPMN нотації (Business Process Model and Notation (BPMN))

2.2.1 Опис техніки BPMN моделювання

BPMN надає стандартну мову для моделювання бізнес-процесів, яка доступна як бізнес-користувачам, так і технічним розробникам. BPMN розроблена для охоплення різних типів моделювання, включаючи як внутрішні (приватні) процеси, так і спільні (публічні) процеси. Вона може слугувати вихідними даними для технологій автоматизації процесів [1].

Ключовою особливістю BPMN є здатність розрізняти діяльність різних учасників процесу за допомогою "пулів" і "доріжок" (lanes). На рис. 2.4 наведено приклад діграми з [1]. Коли потік роботи перетинає межі доріжки, відповідальність за виконання роботи переходить до іншої ролі в організації. Самі доріжки є частиною пулу. Пул – це самостійна бізнес-одиниця (незалежний елемент), зазвичай організація або система. Один пул може включати кілька доріжок, кожна з яких представляє певну роль. Зазвичай процес включає один пул для замовника та другий пул для організації, яку досліджують, хоча можливо включити будь-яку кількість пулів [5].

Моделювання бізнес-процесів за допомогою BPMN допомагає керівни-

кам аналізувати та приймати стратегічні рішення. Це дозволяє їм легше оцінювати вплив змін у бізнес-процесах на різні аспекти компанії та виявляти можливості для автоматизації [6].

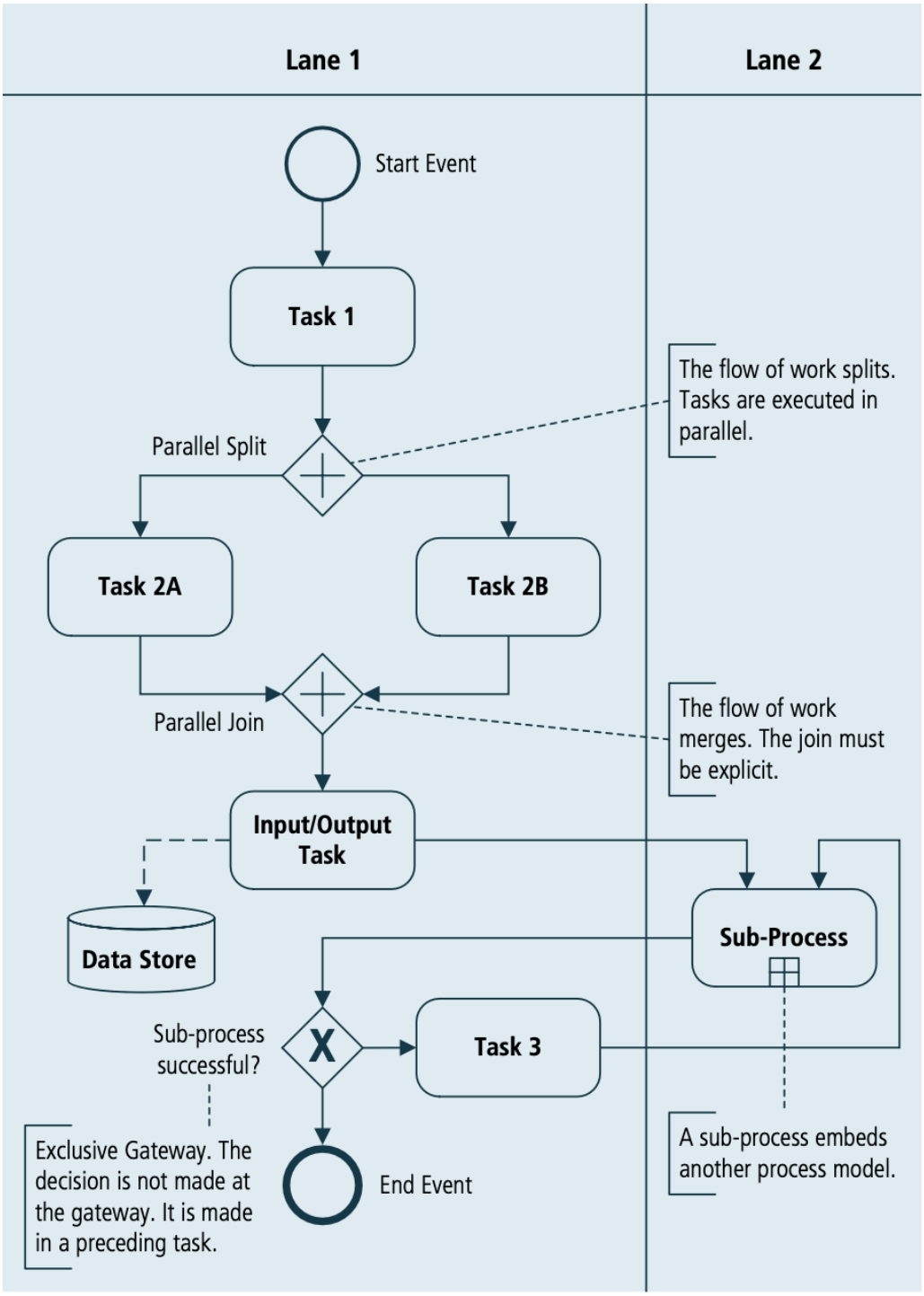

Рисунок 2.4 – Приклад BPMN діаграми

2.2.2 Особливості використання BPMN нотації

Сильні сторони (плюси) використання BPMN нотації [7, 8]:

– задовольняє базове людське розуміння послідовних дій;

– більшість зацікавлених сторін комфортно відчувають себе з концепціями та основними елементами моделі процесу;

– використання різних рівнів дозволяє враховувати різні перспективи різних груп зацікавлених сторін;

– ефективно демонструє, як впоратися з великою кількістю сценаріїв та паралельних гілок;

– допомагає виявити групи зацікавлених сторін, які могли бути проігноровані;

– сприяє виявленню потенційних поліпшень, підкреслюючи "болючі точки" в структурі процесу (тобто візуалізація процесу);

– моделі можуть мати самостійне значення. Вони надають документацію для цілей відповідності і можуть бути використані бізнес-учасниками для навчання та координації діяльності;

– можуть слугувати базою для безперервного вдосконалення;

– забезпечують узгодженість у позначеннях між артефактами;

– надають прозорість і ясність для власників процесів і учасників щодо відповідальності за дії, їх послідовності та передачі обов'язків.

Слабкі сторони (мінуси) використання BPMN нотації:

– для багатьох людей у сфері IT формальна модель процесу часто відображає старіший та більш документозалежний підхід до розробки програмного забезпечення, тому час проєкту не виділяється на розробку моделі процесу, особливо для поточного стану чи проблемної області;

– може стати надзвичайно складною та незручною, якщо не структурована належним чином і це особливо стосується випадків, коли бізнес-правила та рішення не управляються окремо від процесу;

– складні процеси можуть включати безліч дій і ролей, що може усклад-

нити їх розуміння та схвалення однією особою;

– проблеми в процесі не завжди можна виявити, лише переглядаючи модель на високому рівні і зазвичай потрібна більш детальна модель з посиланням на метадані (такі як частота шляхів, витрати та часові фактори);

– у динамічному середовищі, де все швидко змінюється, моделі процесів можуть швидко ставати застарілими;

– може бути важко підтримувати, якщо модель процесу слугує лише документацією, оскільки учасники можуть змінювати процес відповідно до своїх потреб без оновлення моделі.

2.3 SWOT-аналіз (SWOT Analysis)

2.3.1 Опис техніки SWOT-аналіз

SWOT-аналіз – це простий, але ефективний інструмент, який використовується для оцінки сильних і слабких сторін, можливостей та загроз організації в контексті як внутрішніх, так і зовнішніх умов.

Техніка допомагає визначити загальний стан організації як зсередини, так і ззовні. Мова, що використовується в SWOT-аналізі [1], є лаконічною, конкретною, реалістичною та підкріпленою доказами. Інструмент слугує оцінкою організації за визначеними факторами успіху. Його можна проводити на будь-якому рівні: від підприємства в цілому до підрозділу, бізнес-одиниці, проєкту або навіть окремої особи. Виконуючи SWOT-аналіз у структурований спосіб, учасники можуть отримати чіткіше розуміння впливу існуючого набору умов на майбутні [9].

SWOT-аналіз можна використовувати для [7, 10]:

– оцінки поточного середовища організації;

– обміну отриманою інформацією з зацікавленими сторонами;

– визначення найкращих варіантів для задоволення потреб організації;

– виявлення потенційних бар'єрів для успіху та створення планів дій для їх подолання;

– коригування та уточнення планів протягом проєкту в міру виникнення нових потреб;

– виявлення областей сили, які допоможуть організації впроваджувати нові стратегії;

– розробки критеріїв для оцінки успішності проєкту на основі певних вимог;

– виявлення слабких сторін, які можуть підривати цілі проєкту, та розробки стратегій для вирішення актуальних загроз.

SWOT є абревіатурою для Сильних сторін (Strengths (S)), Слабкостей (Weaknesses (W)), Можливостей (Opportunities (O)) і Загроз (Threats (T)) [7]:

– сильні сторони (S) – все, що добре виконує оцінювана група – це можуть бути досвідчені співробітники, ефективні процеси, IT-системи, відносини з клієнтами або будь-який інший внутрішній фактор, що веде до успіху;

– слабкості (W) – дії або функції, які оцінювана група виконує погано або зовсім не виконує;

– можливості (O) – зовнішні фактори, які оцінювана група може використати на свою користь – це можуть бути нові ринки, нові технології, зміни в конкурентному середовищі або інші сили;

– загрози (T) – зовнішні фактори, які можуть негативно вплинути на оцінювану групу – сюди можуть входити такі фактори, як вихід на ринок нового конкурента, економічні спади або інші сили.

Початок SWOT-аналізу з можливостей і загроз задає контекст для виявлення сильних і слабких сторін.

Як практичний приклад використання техніки SWOT-аналізу було взято кейс компанії Starbucks.

Starbucks – це всесвітньо відома мережа кав'ярень, яка пропонує не лише широкий вибір кавових напоїв, а й різноманітні страви та фірмові товари через свою величезну мережу магазинів [11].

Сильні сторони (S):

– сильна глобальна репутація бренду та лояльність клієнтів;

– різноманітний асортимент продукції, що виходить за межі кави.

Слабкості (W):

– високі ціни порівняно з конкурентами;

– чутливість до коливань цін на кавові зерна.

Можливості (O):

– випуск більшої кількості рослинних та здорових продуктів харчування;

– реалізація цифрових ініціатив для покращення процесу замовлення та програм лояльності.

Загрози (T):

– конкуренція з боку місцевих кав'ярень і великих мереж;

– зміна споживчих вподобань на користь здоровіших і стійкіших варіантів.

Щоб залишатися конкурентоспроможними за ціною, Starbucks зосередилася на покращенні клієнтського досвіду через програми лояльності та продовжувала розширення. Вони врахували уподобання споживачів, впроваджуючи рослинні та здорові варіанти харчування.

У відповідь на загрози з боку місцевих кав'ярень і великих мереж Starbucks розширилася на глобальному рівні та зробила акцент на сталий розвиток. Орієнтований на клієнта підхід і стратегії глобального розширення дозволили їм залишатися домінуючою силою в конкурентній кавовій індустрії.

2.3.2 Особливості використання SWOT-аналізу

Сильні сторони (плюси) використання SWOT-аналізу [12, 13]:

– зрозумілість: SWOT-аналіз є простим у виконанні та розумінні, що робить його доступним для всіх учасників процесу;

– широкий спектр застосування: можна використовувати в різних кон-

текстах, від оцінки проєкту до стратегічного планування на рівні підприємства;

– інформаційна вартість: допомагає виявити внутрішні переваги, які можуть бути використані для розвитку нових стратегій або поліпшення існуючих.

Слабкі сторони (мінуси) використання SWOT-аналізу [12, 13]:

– суб'єктивність: результати SWOT-аналізу можуть залежати від думок учасників, що може призвести до упередженості;

– недостатня деталізація: SWOT-аналіз може бути недостатньо глибоким для складних питань, що вимагають детального аналізу;

– непостійність: зовнішні умови можуть швидко змінюватися, і результати SWOT-аналізу можуть стати застарілими.

Висновки за розділом 2

У розділі було розглянуто низку технік, які зазвичай використовуються в IT-компаніях для прийняття рішень в плануванні та реалізації різноманітних проєктів, а також сетапу процесів у компаніях. Серед них було виділено три основних та найбільш застосовних на практиці в сучасному світі – BPMN Моделювання, Моделювання рішень та SWOT-аналіз.

Як результат дослідження, у якості обраного інструменту було взято техніку «Моделювання рішень». Ця техніка є ефективним інструментом для аналізу та оптимізації процесів прийняття рішень в IT-проєктах протягом всього життєвого циклу системи (SDLC). Вона забезпечує візуалізацію бізнес-процесів та правил, що значно спрощує взаємодію між зацікавленими сторонами. Ця техніка дозволяє групувати правила, що стосуються рішень, що сприяє автоматизації та полегшує виявлення вузьких місць у процесах.

Однією з ключових переваг «Моделювання рішень» є те, що воно дозволяє зберігати та представляти знання у структурованому вигляді. Наприклад, моделі можуть бути представлені у формі таблиць або дерев, що робить їх зручними для аналізу та подальшого використання. Вони також підтримують кіль-

ка перспектив, що допомагає виявити потреби різних груп зацікавлених сторін. Також варто зазначити, що «моделювання рішень» може бути використане для вирішення як простих, так і складних бізнес-проблем. Завдяки своїй структурі, ця техніка дозволяє чітко визначати вхідні дані та правила, які необхідні для прийняття рішень, що робить її універсальним інструментом для ІТ-проєктів.

При правильному застосуванні, «моделювання рішень» значно полегшує процес прийняття рішень та підвищує ефективність управління. Для досліджуваної задачі в автоматизації процесів, ця техніка виглядає найбільш привабливою і універсальною. На основі її підвиду «Дерева рішень» будемо вибудовувати шаблон для досягнення мети роботи.

3 ПРОЦЕС ПРИЙНЯТТЯ РІШЕНЬ ВПРОДОВЖ ЖИТТЄВОГО ЦИКЛУ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Основні етапи життєвого циклу розробки програмного забезпечення (SDLC) в аутсорсових IT-компаніях

Життєвий цикл розробки програмного забезпечення в аутсорсових IT-компаніях України має свої особливості, що визначаються потребами глобального ринку, високим рівнем професійності спеціалістів та специфікою співпраці з іноземними клієнтами. Українські IT-компанії відомі своєю клієнтоорієнтованістю, що дозволяє їм успішно працювати з замовниками з різних країн, зокрема США, Європи та Канади. Така співпраця вимагає адаптації до культурних особливостей, часових поясів та індивідуальних бізнес-процесів клієнтів. Гнучкість та вміння швидко реагувати на зміни вимагають застосування сучасних методологій розробки, таких як Agile та Scrum, які допомагають мінімізувати ризики і забезпечують ефективну комунікацію між усіма учасниками проекту.

Процес бізнес-аналізу відіграє важливу роль у початкових етапах життєвого циклу розробки. Саме на цьому етапі відбувається глибокий аналіз вимог замовника, створення документації, яка визначає рамки і напрямки роботи команди. Бізнес-аналітики в Україні приділяють особливу увагу візуалізації процесів за допомогою таких інструментів, як UML-діаграми, BPMN-моделі, а також Decision modeling таблиці та дерева. Це забезпечує прозорість та розуміння проекту на всіх рівнях.

Наші IT-компанії відомі своїм оптимальним співвідношенням якості та вартості. Висококваліфіковані розробники пропонують послуги за нижчими цінами порівняно з країнами Західної Європи чи США, що робить їх привабливими для іноземних клієнтів. Однак, особливою перевагою є не лише вартість, а й якість виконання, яка проявляється у дотриманні міжнародних стандартів та впровадженні інноваційних технологій. Окремою особливістю аутсорсових IT-компаній в Україні є робота у розподілених командах. Це вимагає налагодже-

них процесів комунікації за допомогою сучасних інструментів, таких як Zoom, Slack чи Jira. Крім того, компанії виконують проекти в різних доменах, включаючи фінанси, охорону здоров'я, e-commerce та штучний інтелект. В умовах глобальної конкуренції вітчизняні IT-компанії продовжують вдосконалювати свої процеси, інтегруючи новітні технології та інструменти у життєвий цикл розробки. Їхні команди демонструють не лише технічну компетентність, а й креативність, яка дозволяє пропонувати інноваційні рішення для складних завдань. Такий підхід забезпечує їх конкурентоспроможність на світовому ринку та робить їх надійними партнерами для іноземних замовників.

Процес розробки зазвичай починається з узгодження вимог із замовником, складання контракту та чіткого визначення функціоналу. Планування проєкту включає оцінку ресурсів, часових рамок та бюджету. На етапі розробки команда зазвичай створює MVP (мінімально життєздатний продукт), що дозволяє швидко представити замовнику основний функціонал. Наступним кроком є тестування, яке часто включає автоматизовані методи, що підвищують ефективність і скорочують час. Після впровадження продукту компанії зазвичай надають послуги з підтримки та оновлення. Тож як висновок, можна виділити наступні етапи розробки ПЗ:

– аналіз вимог (Requirements Analysis);
– проєктування (Design);
– розробка і тестування (Development and Testing);
– впровадження та супровід (Deployment and Maintenance).

3.2 Визначення ключових точок прийняття рішень у процесах розробки програмного забезпечення

Управління проєктами в аутсорсових IT-компаніях передбачає постійне прийняття рішень на всіх етапах життєвого циклу розробки програмного забезпечення (SDLC). Успіх проєкту значною мірою залежить від того, наскільки об-

ґрунтовано та своєчасно приймаються ці рішення. У кожному етапі SDLC існують критичні точки, де необхідно аналізувати інформацію, оцінювати альтернативи та обирати оптимальні варіанти дій. Ці точки прийняття рішень, або decision points, є ключовими для забезпечення якості продукту, дотримання строків і бюджету.

У процесі розробки програмного забезпечення рішення приймаються на стратегічному, тактичному та оперативному рівнях. На стратегічному рівні визначаються загальні цілі проєкту, його архітектура, ключові технології та методології. На тактичному рівні рішення стосуються організації робочих процесів і управління командою. Оперативний рівень охоплює прийняття рішень під час вирішення конкретних задач, таких як налаштування середовища розробки або вирішення конфліктів у коді.

Розглянемо, які ключові точки прийняття рішень є найбільш важливими на різних етапах циклу розробки.

1. Аналіз вимог (Requirements Analysis). На цьому етапі вирішується, які функціональні можливості будуть включені до продукту, як вони задовольнять потреби замовника та як врахувати всі бізнес-обмеження. Рішення на цьому етапі впливають на весь подальший цикл розробки.

Ключові рішення:

– як розставити пріоритети між різними вимогами клієнта;

– чи потрібно уточнювати або доповнювати вимоги;

– як уникнути можливих суперечностей або пропусків у вимогах.

2. Проєктування (Design). У процесі проєктування приймаються рішення про архітектуру, технологічний стек і візуалізацію процесів. Ці рішення визначають, наскільки добре продукт відповідатиме вимогам клієнта і наскільки легко його буде підтримувати та масштабувати.

Ключові рішення:

– чи доцільно використовувати існуючі шаблони архітектури або створювати нові;

– як забезпечити зручність та ефективність інтеграції майбутнього проду-

кту з іншими системами;

– який рівень деталізації діаграм і моделей обрати для оптимальної комунікації між членами команди.

3. Розробка (Development). На етапі розробки рішення часто приймаються в умовах обмеженого часу. Ефективність розробки залежить від того, наскільки добре команда взаємодіє між собою та наскільки злагоджено впроваджуються рішення, ухвалені на попередніх етапах.

Ключові рішення:

– яким чином розподілити завдання між членами команди;

– чи потрібно змінювати пріоритети під час виконання проєкту;

– як оперативно реагувати на нові вимоги клієнта.

4. Тестування (Testing). У процесі тестування приймаються рішення, які дозволяють мінімізувати ризики та забезпечити випуск продукту високої якості. Ці рішення включають вибір стратегій тестування та визначення, які сценарії є критичними для успіху продукту.

Ключові рішення:

– як організувати процес тестування: ручне чи автоматизоване;

– які тестові сценарії виконати першочергово;

– як забезпечити баланс між швидкістю тестування та його глибиною.

5. Впровадження (Deployment). На етапі впровадження важливими є рішення, які забезпечують безперебійну інтеграцію продукту у продуктивне середовище замовника. Тут також враховується ризик виникнення проблем під час запуску.

Ключові рішення:

– як побудувати план впровадження: поступовий або одразу для всіх користувачів;

– як організувати резервне копіювання для забезпечення безпеки;

– чи потрібна попередня підготовка користувачів до роботи з продуктом.

6. Супровід (Maintenance). Підтримка продукту вимагає постійного прийняття рішень щодо його вдосконалення, усунення помилок і адаптації до змін у

бізнес-середовищі замовника.

Ключові рішення:

– які оновлення є найбільш пріоритетними;

– як підготувати продукт до нових викликів на ринку;

– чи потрібна інтеграція з новими технологіями.

Для аутсорсових IT-компаній, де важливо забезпечити прозорість і контроль процесів, формалізація точок прийняття рішень є необхідною. Використання таких технік, як Decision Tree, дозволяє не лише структурувати процес ухвалення рішень, а й зменшити ризики, пов’язані з людським фактором. Формалізація сприяє підвищенню ефективності команди та покращенню комунікації між її членами.

3.3 Вплив бізнес-аналітика на процес прийняття рішень впродовж життєвого циклу розробки програмного забезпечення

Автоматизація прийняття рішень є ключовим інструментом бізнес-аналітика на етапах аналізу вимог та проєктування, особливо у контексті використання техніки моделювання рішень, такої як Decision Tree (дерево рішень). Ця техніка дозволяє систематизувати процес прийняття рішень шляхом візуалізації варіантів і наслідків, що є надзвичайно корисним для складних проєктів, де потрібно враховувати численні змінні та сценарії.

На етапі аналізу вимог основна роль бізнес-аналітика полягає у зборі, інтерпретації та структуруванні потреб замовника. Використання дерева рішень у цьому контексті дозволяє створити чітку і зрозумілу модель, яка демонструє, як різні фактори впливають на вибір конкретного рішення. Наприклад, під час визначення функціоналу програмного забезпечення аналітик може використовувати дерево рішень для оцінки варіантів впровадження певної функції, враховуючи ресурси, технічні обмеження та бізнес-пріоритети. Завдяки цьому модель слугує ефективним засобом комунікації між командою розробників і замо-

вником.

На етапі проєктування дерева рішень також допомагають структурувати процес вибору архітектурних рішень, які відповідають вимогам замовника. Моделі дозволяють бізнес-аналітикам і технічним фахівцям оцінювати наслідки кожного варіанту, враховуючи їхній вплив на вартість, час розробки та ризики. Це особливо важливо для проєктів з обмеженими ресурсами, де помилки на початкових етапах можуть призвести до значних витрат у майбутньому.

Хоча решта етапів життєвого циклу розробки ПЗ, таких як розробка, тестування та впровадження, більше належать до зони компетентності інших спеціалістів (розробників, тестувальників, DevOps інженерів), рішення, прийняті на етапах аналізу та проєктування, визначають успішність цих етапів. Саме тому автоматизація прийняття рішень бізнес-аналітиком є критичною для забезпечення ефективності всього процесу.

Однією з ключових переваг автоматизації є можливість стандартизації процесів. Розробка універсальних темплейтів дерев рішень на рівні компанії для всіх нових проєктів є стратегічно важливим кроком. Такі темплейти можуть включати типові сценарії, що охоплюють найпоширеніші завдання, з якими стикається компанія. Наприклад, можна створити дерева рішень для оцінки ризиків, визначення пріоритетів функціоналу чи вибору оптимального підходу до інтеграції зовнішніх систем.

Шаблони мають такі переваги:

– зниження часу на аналіз: бізнес-аналітики отримують можливість швидко адаптувати готові рішення до специфіки нового проєкту;

– уніфікація підходів: використання спільних темплейтів гарантує однаковий рівень якості аналізу для всіх проєктів, що сприяє підвищенню ефективності команди;

– спрощення навчання: нові працівники легше інтегруються у робочий процес, використовуючи існуючі шаблони як базу для навчання;

– підвищення прозорості: замовники та інші учасники проєкту можуть легко зрозуміти логіку прийнятих рішень завдяки уніфікованій візуалізації.

У довгостроковій перспективі розробка та використання темплейтів дерев рішень сприяє оптимізації процесу управління проєктами, підвищує якість обслуговування клієнтів та знижує ризики, пов'язані з неефективними або невчасними рішеннями. Для компаній, які активно працюють в аутсорсовій моделі, це стає значною конкурентною перевагою на глобальному ринку.

3.4 Аналіз інструментів побудови дерев прийняття рішень

Для побудови Decision Tree (дерева рішень) існує безліч інструментів, які можуть бути корисними в різних сценаріях аналізу та моделювання. Вибір інструменту залежить від конкретних потреб, складності проєкту, доступних ресурсів і рівня підготовки команди та аналітика. Розглянемо найбільш відомі і популярні рішення, що використовуються в більшості IT компаній України.

1. Microsoft Excel. Excel пропонує простий і доступний спосіб створення Decision Trees через функції діаграм або додаткові інструменти, такі як "TreePlan". Інтерфейс базується на добре знайомих таблицях і панелях управління. Для створення дерева потрібно вручну вводити вузли та рішення у вигляді схеми. Інструмент простий у використанні, але обмежений в обробці великих наборів даних і вимагає багато ручної роботи.

Плюси:

– доступність: майже всі компанії мають доступ до Excel;

– простота використання для базових дерев рішень;

– вбудовані функції для формул, таблиць і діаграм, які можна адаптувати для створення дерева.

Мінуси:

– обмежені можливості для складних моделей;

– відсутність інтеграції з іншими системами;

– потреба в ручному налаштуванні та оновленні.

2. Lucidchart. Lucidchart – це веб-інструмент для створення діаграм і дерев

рішень. Інтерфейс побудований за принципом drag-and-drop, дозволяючи користувачам легко додавати вузли, гілки та підписи. Інструмент пропонує шаблони, що економлять час, і можливість спільного редагування в режимі реального часу. Lucidchart особливо зручний для командної роботи, але обмежена кількість функцій доступна у безкоштовній версії.

Плюси:

– простий інтерфейс із широким набором шаблонів;

– підтримка спільної роботи в режимі реального часу;

– інтеграція з Google Workspace, Microsoft Office, Slack тощо.

Мінуси:

– потребує передплати для доступу до розширених функцій;

– менш зручний для аналізу великих обсягів даних.

3. IBM SPSS Decision Trees. IBM SPSS пропонує інтерактивний інтерфейс, орієнтований на аналіз даних. Користувач завантажує дані, вибирає модель дерева (наприклад, CART) і налаштовує параметри через графічний інтерфейс. Дерева автоматично створюються і візуалізуються, що полегшує аналіз складних наборів даних. SPSS підходить для корпоративних користувачів, але потребує ліцензії, що може бути дорогим варіантом.

Плюси:

– потужний інструмент для побудови дерев рішень, що підтримує складні алгоритми та статистичний аналіз;

– можливість інтеграції з іншими продуктами IBM SPSS;

– відмінний для обробки великих даних.

Мінуси:

– висока вартість ліцензії;

– складність освоєння для початківців.

4. Python (sklearn, graphviz, dtreeviz). Python, у поєднанні з бібліотекою sklearn (Scikit-learn), дозволяє створювати Decision Trees програмним шляхом. Інтерфейс базується на коді, тому необхідні навички програмування. Побудова дерева включає завантаження даних, тренування моделі через функцію

DecisionTreeClassifier або DecisionTreeRegressor і подальшу візуалізацію через бібліотеки, такі як matplotlib або graphviz. Інструмент підходить для аналізу великих наборів даних і автоматизації побудови моделей. Інтерфейс може бути менш зручним для візуального редагування, але його потужність у роботі з великими обсягами даних є беззаперечною.

Плюси:

– повна гнучкість і можливість автоматизації;

– бібліотеки, такі як sklearn, дозволяють швидко побудувати дерево рішень на основі даних;

– візуалізація за допомогою Graphviz або dtreeviz забезпечує професійний вигляд;

– відкритий код, безкоштовність.

Мінуси:

– потрібні навички програмування;

– триваліший процес навчання порівняно з візуальними інструментами.

5. R (rpart, partykit). R використовує бібліотеки, такі як rpart і caret, для побудови дерев рішень. Користувач вводить дані через кодовий інтерфейс, створюючи дерево за допомогою функцій, таких як rpart(). Візуалізація здійснюється через такі інструменти, як rpart.plot. Інтерфейс повністю текстовий, і користувачі мають вручну налаштовувати параметри дерева. Програмний підхід дозволяє гнучко адаптувати дерево до складних сценаріїв аналізу, але вимагає розуміння мови R.

Плюси:

– потужна статистична підтримка;

– широкий вибір пакетів для побудови та аналізу дерев;

– відмінний для дослідницького аналізу даних.

Мінуси:

– складність у використанні для тих, хто не знайомий з R;

– відсутність інтерактивного графічного інтерфейсу.

6. KNIME Analytics Platform. KNIME працює через візуальне моделюван-

ня робочого процесу. Користувач додає модулі (вузли), наприклад, для обробки даних, побудови дерева та аналізу результатів, з'єднуючи їх у єдиний робочий ланцюг. Інтерфейс інтерактивний, з великою кількістю опцій для налаштування. KNIME підходить для візуального програмування і не вимагає глибоких навичок кодування, але потребує початкового навчання.

Плюси:

– інтуїтивний інтерфейс із підтримкою побудови дерев за допомогою drag-and-drop;

– інтеграція з іншими платформами для аналізу даних;

– можливість роботи з великими обсягами даних.

Мінуси:

– потребує налаштування для складних дерев;

– певна крива навчання для новачків.

7. XMind. XMind використовується для побудови дерев у вигляді ментальних карт. Інтерфейс інтуїтивний, з фокусом на дизайні й організації ідей. Користувач створює дерево, додаючи вузли та підвузли, які автоматично формуються у структуровану діаграму. XMind ідеально підходить для презентацій або планування, але має обмежені можливості для інтеграції з іншими інструментами.

Плюси:

– можливість інтерактивної візуалізації дерев рішень;

– підтримка інтеграції з базами даних та іншими джерелами;

– простота у створенні звітів і дашбордів.

Мінуси:

– фокус на візуалізації, а не на аналітичних алгоритмах;

– висока вартість для команди.

8. Draw.io (тепер відоме як diagrams.net). Draw.io – це простий і безкоштовний інструмент для створення діаграм. Інтерфейс базується на принципі drag-and-drop: користувачі додають вузли, з'єднують їх стрілками та налаштовують стилі. Draw.io пропонує велику кількість форм і шаблонів, що робить йо-

го універсальним для створення дерев рішень. Його основна перевага – простота і доступність, хоча він не підходить для складного аналізу даних або інтеграції з програмним забезпеченням.

Плюси:

– безкоштовність: Draw.io є безкоштовним і доступним онлайн-інструментом;

– простота використання: інтуїтивно зрозумілий інтерфейс з функцією drag-and-drop для створення діаграм;

– інтеграція з Google Drive та іншими хмарними платформами, що дозволяє зберігати і працювати з діаграмами в хмарі, що полегшує спільну роботу;

– широкий вибір шаблонів: Draw.io надає різноманітні шаблони для побудови різних типів діаграм, включаючи Decision Tree;

– безпечність: діаграми можна зберігати локально або в хмарі, що забезпечує гнучкість у збереженні даних.

Мінуси:

– обмежені можливості для складних аналізів: Draw.io не має вбудованих функцій для автоматичного аналізу або побудови моделей з великими даними, як Python або R;

– менш підходить для аналізу великих даних: ідеально підходить для створення графічних представлень, але не для комплексного обчислення результатів;

– відсутність інтеграції з аналітичними інструментами: для більш складних дерев рішень чи аналізу даних потрібні додаткові інструменти.

Після аналізу і порівняння інструментів по популярності, можна зробити наступний висновок:

– Draw.io є дуже популярним завдяки своїй безкоштовності та простоті використання, особливо серед малих та середніх компаній, що використовують діаграми для візуалізації процесів;

– Python (sklearn) та R залишаються найбільш відповідними для розробки складних моделей і обробки великих обсягів даних;

– Lucidchart і Microsoft Excel зручні для користувачів, яким необхідна простота в інтеграції з іншими інструментами та швидкість створення діаграм;

– IBM SPSS і KNIME підходять для більш складних, аналітичних задач у великих організаціях.

Відповідно, Draw.io є відмінним вибором для створення простих, зрозумілих дерев рішень та інших діаграм, що легко інтегруються в робочі процеси команди. Якщо ж необхідно поєднати глибокий аналіз і візуалізацію з даними, то для більш складних моделей краще використовувати Python або R. Draw.io найкраще підходить для бізнес-аналітиків та менеджерів, яким необхідно швидко і доступно візуалізувати рішення, не маючи необхідності у складних алгоритмах або аналітиці.

3.5 Приклад дерева рішень

Розглянемо приклад дерева рішень [14].

Компанія Really Big Ideas має тримісячне вікно можливостей для розробки нового продукту з використанням нещодавно створеного програмного забезпечення для розпізнавання патернів. Програмне забезпечення легко адаптується до різних застосувань. Компанія має ресурси та час для розробки лише одного з двох проєктів або може відмовитися від розробки.

Детектор диму та вогню коштуватиме $100 000 на розробку, і, якщо він буде успішним, відділ бізнес-аналізу прогнозує дохід у розмірі $1 000 000.

Детектор руху, який використовує звичайне домашнє освітлення, коштуватиме лише $10 000 на розробку. Аналітики очікують, що такий пристрій принесе $400 000 доходу.

Ймовірність успіху детектора диму та вогню складає 50%, тоді як детектор руху має 80% шансів на успіх.

Наше завдання: допомогти компанії Really Big Ideas вирішити, який продукт варто розробляти, якщо взагалі варто. Потрібно оцінити альтернативи кі-

лькісно, враховуючи наявні невизначеності.

Створимо дерево рішень. Дерево рішень – це діаграма, яка складається з вузлів та гілок, що їх з'єднують. Вузли позначають точки прийняття рішень, випадкові події або кінцеві гілки. Гілки відповідають кожній альтернативі рішення або результату події, які виходять із вузла.

Перше рішення (кореневий вузол). Для спрощення розв’язання задачі було вибрано draw.io як інструмент візуалізації. Починаємо із малювання невеликого квадрата з лівого боку нашого середовища. Цей квадрат називається кореневим вузлом або коренем. Кореневий вузол представляє перший набір альтернатив рішень.

Для кожної альтернативи рішення малюємо лінію або гілку, що простягається вправо від кореневого вузла. Залишаємо достатньо простору між лініями, щоб додати інформацію. Деякі гілки можуть розгалужуватись у додаткові альтернативи рішень або результати. Можна "згинати" гілки, щоб вирівняти їх по горизонталі. Такі техніки допомагають легше відстежувати альтернативи (рис. 3.1).

Позначимо кожну гілку відповідною альтернативою рішення та пов’язаними інвестиційними витратами. Зазначаємо, що розробка детектора диму та вогню коштуватиме 100 000 доларів. Аналогічно, вказуємо, що розробка детектора руху коштуватиме 10 000 доларів. Пишемо $0 для третьої гілки, яка відповідає альтернативі не розробляти жоден із продуктів.

У сценарії Really Big Ideas кожна спроба розробки продукту може мати один із двох результатів: проєкт може або стати успішним, або зазнати невдачі. Малюємо невелике коло, яке називається вузлом випадковості (chance node), в кінці гілки для детектора диму та вогню. Також малюємо вузол випадковості в кінці гілки для детектора руху. Від кожного вузла випадковості проводимо дві гілки вправо; одна гілка представляє успіх, а інша – невдачу. Позначаємо гілки Success/Failure відповідно на рис. 3.2.

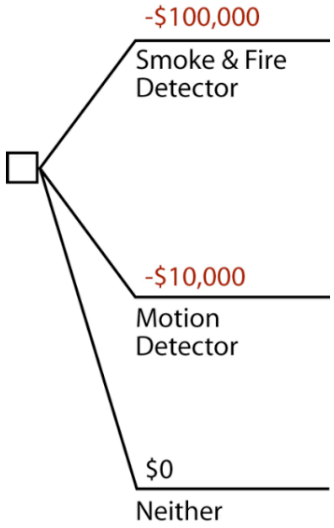

Рисунок 3.1 – Дерево рішень (перше рішення)

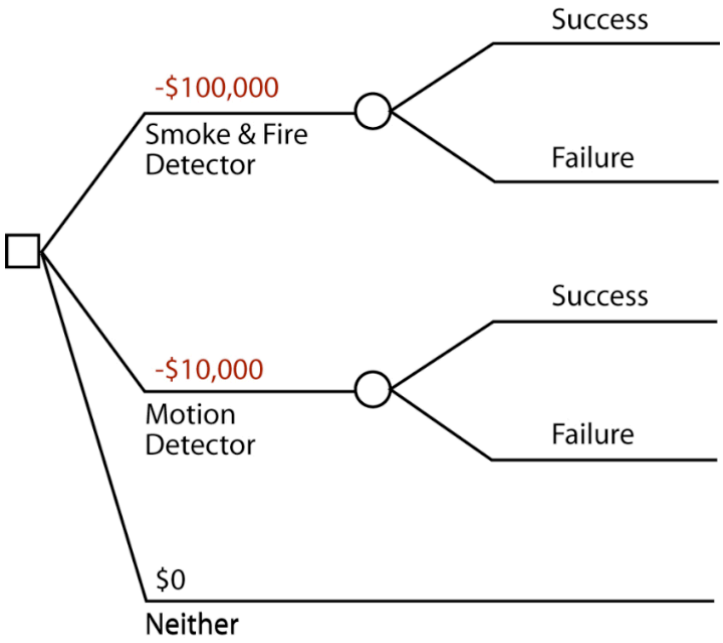

Рисунок 3.2 – Дерево рішень (випадкові результати)

Тепер можна завершити всі гілки, додаючи кінцеві вузли, оскільки подальша інформація для розгалуження відсутня. Намалюємо невеликий трикутник на кінці кожної гілки, що представляє кінцевий вузол. Біля кожного вузла вказуємо значення прибутку (payoff). У бізнес-застосунках прибуток зазвичай виражається у грошовому еквіваленті, що дорівнює очікуваному чистому прибутку або доходу на інвестиції.

Чистий прибуток (або збиток) – це різниця між вартістю інвестицій і за-

гальним доходом. Позитивне значення вказує на чистий прибуток, тоді як негативне значення – на чистий збиток. Іншими словами, якщо дохід перевищує інвестиції, то проєкт є прибутковим. В іншому випадку результатом є збиток або, якщо прибуток дорівнює нулю, результат "без втрат".

Для компанії Really Big Ideas успішний проєкт із розробки детектора диму та вогню принесе валовий дохід у розмірі $1 000 000. Відповідний чистий прибуток розраховується як сума валового доходу та вартості інвестицій. Враховуючи, що вартість інвестицій можна представити як від'ємне число, обчислення виглядає так:

$$\$1000000 + (-\$100000) = \$900000 \text{ чистого прибутку.}$$

Тож пишемо $900 000 на кінці гілки, яка представляє успіх детектора диму та вогню.

Якщо ж проєкт із розробки детектора диму та вогню буде невдалим, то жодного доходу не буде отримано, і всі інвестиції будуть втрачені. Розрахунок для цього випадку:

$$\$0 + (-\$100000) = -\$100000,$$

що є збитком або негативним значенням прибутку. Вказуємо $-\$100000$ на кінці гілки, яка представляє невдачу детектора диму та вогню.

Проводимо аналогічний розрахунок для успіху та невдачі детектора руху. У результаті успішний проєкт детектора руху матиме прибуток у розмірі $290 000, а у випадку невдачі – збиток $-\$10000$. Пишемо ці значення на кінцях відповідних гілок.

Прибуток для гілки рішення не розробляти жоден із проєктів просто дорівнює $0.

Наведені розрахунки вносимо у вузли дерева (рис. 3.3).

Це завершує базову структуру дерева рішень для альтернатив компанії Really Big Ideas. Тепер ми можемо врахувати ймовірності успіху та невдачі й використати їх для аналізу варіантів рішень.

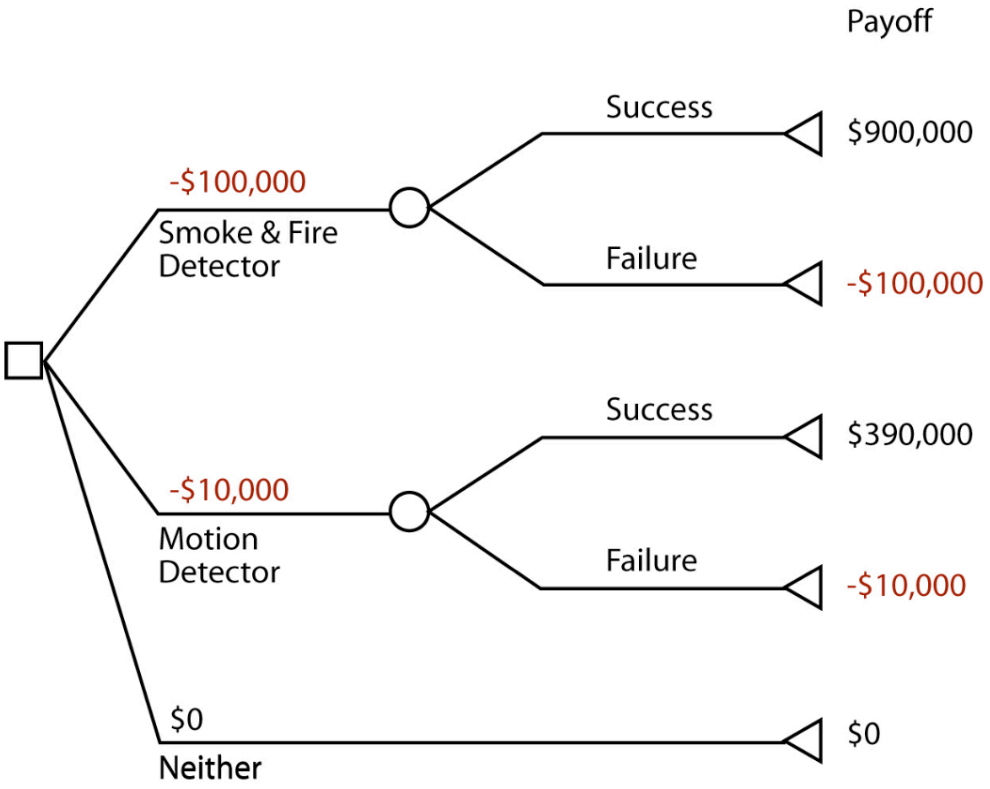

Рисунок 3.3 – Дерево рішень (кінцеві вузли та прибутки)

Тепер можемо врахувати відносну ймовірність результату або невизначеність, пов'язану з кожною випадковою подією. Ймовірності можна виразити у відсотках або десяткових дробах. У цьому прикладі використовується загальноприйнята конвенція дерева рішень, де ймовірності виражаються десятковими дробами від 0,0 до 1,0, де $1{,}0 = 100\%$.

У сценарії Really Big Ideas ймовірність успіху детектора диму та вогню становить 50%, а отже, ймовірність невдачі – також 50%. Тому залишаємо 0,5 на гілці успіху і 0,5 на гілці невдачі. Ймовірність успіху детектора руху становить 80%, або 0,8, а ймовірність невдачі – 20%, або 0,2. Запишемо ці значення на відповідних гілках (рис. 3.4).

Тепер ми готові оцінити відносні переваги кожної альтернативи рішення.

Очікувана вартість (EV) – це спосіб об'єднання прибутків та ймовірностей для кожного вузла. Чим вища EV, тим краща конкретна альтернатива рішення в середньому порівняно з іншими альтернативами в дереві рішень.

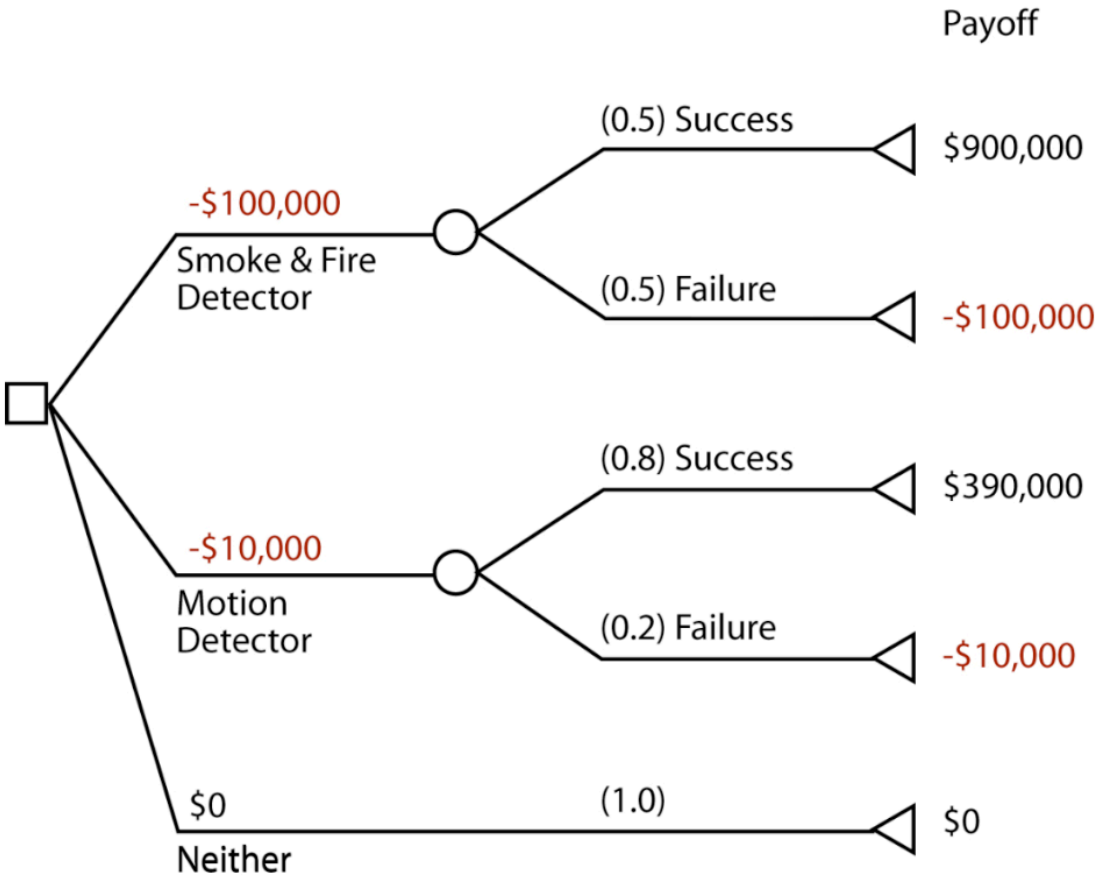

Рисунок 3.4 – Дерево рішень (ймовірність результату)

Розрахуємо EV для будь-якого вузла випадковості, сумуючи EV для кожної гілки, що підключена до цього вузла. Загальна формула для обчислення EV в будь-якому вузлі випадковості виглядає так:

$$EV_{\text{вузол випадковості}} = EV_{\text{гілка}_1} + EV_{\text{гілка}_2} + \ldots + EV_{\text{гілка}_N}.$$

У сценарії Really Big Ideas, якщо детектор диму та вогню успішний, то EV становить прибуток (payoff), помножений на його ймовірність:

$$EV = \$900\,000 \cdot 0{,}5 = \$450\,000.$$

Якщо проєкт детектора диму та вогню зазнає невдачі, то EV буде дорівнювати

$$EV = -\$100\,000 \cdot 0{,}5 \; = -\$50\,000.$$

EV для рішення щодо розробки детектора диму та вогню (враховуючи як успіх, так і невдачу) – це сума EV для всіх можливих результатів:

$$EV_{\text{вузол}} = \left(EV_{\text{успіх}} + EV_{\text{невдача}}\right) = \$450\,000 + (-\$50\,000) \; = \; \$400\,000.$$

Аналогічно, EV для рішення про розробку детектора руху розраховується як:

$$EV = \$390\,000 \cdot 0{,}8 + (-\$10\,000 \cdot 0{,}2) = \$310\,000.$$

Запишемо EV для кожного вузла поруч із цим вузлом (рис 3.5).

Проєкт детектора диму та вогню має вищу EV, ніж детектор руху. Отже, в результаті аналізу виявлено, що:

– детектор диму та вогню є кращим проєктом для розробки, незважаючи на більший ризик; значно більші очікувані прибутки роблять цей ризик більш прийнятним порівняно з конкуренцією;

– детектор руху менш ризикований, але також значно менш прибутковий; з урахуванням очікуваних прибутків цей проєкт не перевищує очікувану вартість проєкту-конкурента.

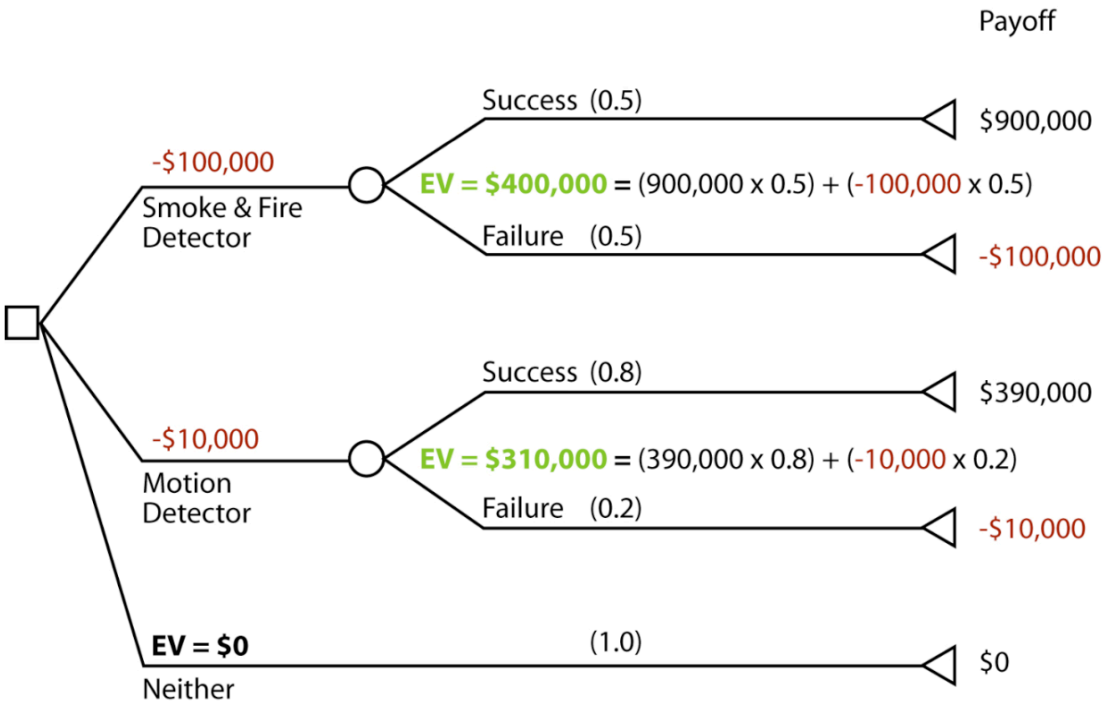

Рисунок 3.5 – Дерево рішень (очікувана вартість)

Висновки за розділом 3

У розділі було розглянуто детальний цикл розробки ПЗ, та визначено ключові точки у ньому, де має сенс автоматизувати систему прийняття рішень з боку бізнес аналітика. Окрім цього, було розглянуто і проаналізовано основні інструменти, за допомогою яких можна візуалізувати розроблені дерева рішень.

Варто зазначити, що автоматизація прийняття рішень за допомогою дерев рішень є ключовим інструментом для бізнес-аналітиків, що дозволяє систематизувати процеси аналізу та проєктування, підвищуючи прозорість, ефективність і якість обслуговування. Використання стандартизованих шаблонів допомагає швидко адаптувати рішення під нові проєкти, знижує ризики, економить час і спрощує інтеграцію нових працівників, що особливо важливо для аутсорсингових компаній на міжнародних ринках. Це сприяє підвищенню довіри клієнтів, оптимізації управління проєктами та зниженню витрат.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ

4.1 Мова програмування Python та її можливості для розробки систем прийняття рішень

Python – це високорівнева мова програмування з відкритим вихідним кодом, яка здобула популярність завдяки простоті синтаксису, широким можливостям і розгалуженій екосистемі бібліотек. Завдяки своїм характеристикам, Python став потужним інструментом для розробки систем прийняття рішень у різних галузях, включаючи бізнес-аналітику, штучний інтелект (ШІ), і обробку даних [15].

Одна з ключових переваг Python – це його бібліотеки та фреймворки, які забезпечують ефективну роботу з алгоритмами прийняття рішень. Наприклад, бібліотека Scikit-learn надає інструменти для реалізації алгоритмів кластеризації, регресії та дерева рішень, що є основою багатьох моделей прийняття рішень. Для роботи з великими наборами даних широко використовуються Pandas і NumPy, які забезпечують швидкий доступ до обробки структурованих і неструктурованих даних. Бібліотека PyTorch або TensorFlow дозволяють створювати глибокі нейронні мережі, що можуть бути інтегровані в системи прийняття рішень для складних сценаріїв.

Python також підтримує візуалізацію даних, що є важливим аспектом аналізу варіантів у системах прийняття рішень. За допомогою бібліотек Matplotlib і Seaborn можна створювати графіки, які відображають ймовірності, очікувані значення та інші показники ефективності рішень. Для інтерактивних панелей управління та візуалізації процесів можна застосовувати Dash або Streamlit [16].

У контексті побудови систем прийняття рішень на етапі Discovery Phase, Python дозволяє бізнес-аналітикам і розробникам створювати інтерактивні моделі для оцінки різних сценаріїв. Наприклад, реалізація дерева рішень може бути виконана за допомогою DecisionTreeClassifier із Scikit-learn або побудови власних функцій для розрахунку очікуваних значень (Expected Value). Це дає

змогу порівнювати різні варіанти реалізації проєктів, обирати найбільш вигідні та мінімізувати ризики.

Загалом, Python забезпечує зручну інтеграцію між аналітикою, моделюванням і автоматизацією прийняття рішень, що робить його ідеальним інструментом для бізнес-аналітиків і розробників, які працюють у сфері оптимізації процесів та аналізу проєктів.

4.2 Алгоритм для подальшої програмної реалізації

Метою роботи є розробка автоматизованого темплейта для бізнес-аналітика саме на фазі аналізу вимог ПЗ, для цього розглянемо більш детальніше процес старту роботи з проєктом [15].

Pre-sale, або discovery фаза, є однією з ключових у процесі розробки IT-продуктів. Вона передує основному циклу реалізації проєкту та служить основою для прийняття рішень щодо його доцільності, структури та стратегії реалізації. Цей етап поєднує бізнес-цілі клієнта з технічними можливостями виконавця, забезпечуючи основу для ефективного планування майбутньої роботи.

Основною метою цієї фази є глибоке розуміння потреб клієнта, визначення цілей проєкту, аналіз ризиків і оцінка технічних, бізнесових та фінансових аспектів. Pre-sale/discovery дозволяє знизити невизначеність і уникнути ризиків, пов'язаних із неправильним плануванням чи виконанням проєкту. Це етап, на якому виконується аналіз ринку, конкурентів, та визначаються ключові вимоги до продукту чи рішення.

Основні задачі бізнес-аналітика в рамках pre-sale/discovery фази:

– збір та аналіз вимог: аналітик проводить зустрічі з клієнтом (workshops, інтерв'ю) для збору початкових вимог, уточнює очікування та цілі бізнесу, а також виявляє проблеми, які має вирішити продукт. та допомагає клієнту сфокусуватися на ключових аспектах, які принесуть максимальну цінність;

– оцінка обсягу роботи: на цьому етапі визначається масштаб проєкту, кі-

лькість необхідних ресурсів, час і бюджет, відбувається складання списку функціональних та нефункціональних вимог;

– визначення ризиків: аналітик аналізує потенційні технологічні, фінансові, організаційні ризики, а також ризики, пов'язані зі зміною ринку чи вимог клієнта, для чого складаються матриці ризиків та план їхньої мінімізації;

– створення roadmap: в результаті роботи формується високорівневий план реалізації проєкту, який включає ключові етапи роботи, орієнтовний час виконання, розподіл завдань та відповідальностей;

– підготовка пропозиції (proposal): за результатами фази формується пропозиція, яка включає технічне рішення, бізнес-кейс, оцінку бюджету та термінів, тобто формується документ, який клієнт використовує для ухвалення рішення про подальшу співпрацю.

Наведемо приклади основних активностей бізнес-аналітика під час pre-sale/discovery фази. Перш за все, на цьому етапі аналітик вивчає галузь, у якій працює клієнт, його бізнес-модель, конкурентів, і проблеми, які потрібно вирішити. Аналітик може використовувати такі методи, як SWOT-аналіз чи інтерв'ю з ключовими стейкхолдерами. Далі, важливим етапом є розуміння кінцевих користувачів. Аналітик визначає їхні потреби, поведінку та проблеми, які продукт має вирішити. Створення персон (personas) дозволяє ефективніше орієнтуватися на потреби кінцевих користувачів.

Коли ідея рішення вже зрозуміла, настає час для визначення дизайну архітектури майбутнього продукту. Аналітик разом із технічними експертами визначає, які технології найкраще підходять для реалізації продукту. Вибір може залежати від масштабованості, бюджету, часу на розробку та технічних вимог.

Після визначення основних вимог і архітектури можна починати розробку технічного рішення, а саме, створення діаграми процесів (BPMN, UML), технічної документації та прототипів. Це дозволяє клієнту побачити, як виглядатиме майбутній продукт і як він буде працювати.

Далі настає етап оцінки, а саме фінансової оцінки. Проводиться оцінка вартості проєкту, враховуючи ресурси, час і ризики. Клієнту пропонуються кі-

лька варіантів бюджету залежно від масштабу робіт і пріоритетів [17].

На основі активностей, описаних вище за текстом, робиться висновок, які можливі рішення можна запропонувати клієнту.

Опція номер один – це розробка MVP (Minimum Viable Product).

MVP – це мінімально життєздатний продукт із базовою функціональністю, який дозволяє протестувати ключові гіпотези та отримати зворотний зв'язок від кінцевих користувачів. Рішення створити MVP приймається, коли у клієнта є обмежений бюджет, короткі терміни або невпевненість щодо повного обсягу вимог. На цьому етапі бізнес-аналітик створює roadmap для визначення мінімальних функцій, які допоможуть досягти основної мети продукту. Наприклад, якщо йдеться про маркетплейс, MVP може включати лише можливість перегляду товарів і додавання до кошика без розширених функцій, як-от рекомендації або складний процес оплати. MVP також дозволяє клієнту протестувати продукт на реальній аудиторії, зібрати аналітику і визначити, чи є сенс розширювати проєкт у майбутньому. Це рішення підходить для стартапів і ситуацій, коли потрібно швидко оцінити ринок.

Другий варіант – це розробки повноцінного проєкту. Це підхід, коли планування охоплює всі етапи: аналіз, дизайн, розробку, тестування, реліз і підтримку. Цей варіант обирають, якщо клієнт має чітко сформовані вимоги, стабільний бюджет і зрозумілі цілі. Бізнес-аналітик на цьому етапі створює детальний план (work breakdown structure), включаючи специфікації, бізнес-вимоги та технічні рішення. Аналітик співпрацює з клієнтом для пріоритизації вимог, проводить сесії з визначення scope проєкту та розробляє backlog із завданнями для команди. Наприклад, якщо мова йде про корпоративну CRM-систему, яка має інтегруватися з існуючими процесами компанії, бізнес-аналітик враховує всі нюанси бізнес-процесів, щоб продукт відповідав очікуванням. У підсумку, реалізація повноцінного проєкту забезпечує високу якість і покриває всі запити клієнта, але вимагає більше часу, ресурсів і ретельного управління.

Наступний варіант – це реалізація PoC (Proof of Concept). Якщо клієнт сумнівається у можливості реалізації складних технологічних функцій, аналітик

може запропонувати створити PoC. Це коротка й обмежена реалізація, яка дозволяє перевірити технічну здійсненність певної ідеї або технології. Наприклад, якщо планується застосунок із використанням штучного інтелекту для розпізнавання зображень, PoC може перевірити, наскільки алгоритм працює з наявними даними. На відміну від MVP, PoC не орієнтована на кінцевих користувачів, а спрямована на внутрішню перевірку концепції [17].

Ну і остання можлива опція – це інтеграція з існуючими продуктами. Замість створення повноцінного нового продукту аналітик може запропонувати інтеграцію нових функцій або модулів до вже існуючого рішення клієнта. Наприклад, якщо клієнт уже має CRM-систему, то можна запропонувати інтеграцію модуля для автоматизації маркетингу замість створення нового окремого продукту. Це економить ресурси і дозволяє уникнути зайвої складності.

Таким чином, ми маємо 4 опцій для старту, плюс опція «не стартувати проєкт взагалі». Це складне, але важливе рішення, яке приймається, якщо аналіз показує, що проєкт не буде рентабельним або його реалізація пов’язана з високими ризиками. Бізнес-аналітик оцінює очікувані витрати, ризики та можливі прибутки, щоб допомогти клієнту уникнути збитків. Наприклад, якщо клієнт хоче створити мобільний застосунок для ринку, який уже насичений аналогами з більш вигідними функціями, є ймовірність, що продукт не отримає популярності. Також аналітик може виявити технологічні обмеження, які унеможливлюють реалізацію проєкту в межах визначеного бюджету. У такому разі аналітик готує звіт із висновками та пропонує альтернативи: наприклад, доопрацювання ідеї або перенесення запуску на більш сприятливий час. Це рішення дозволяє клієнту уникнути марних витрат і сконцентруватися на більш перспективних ініціативах.

Тож як висновок, можна зазначити, що на pre-sale/discovery фазі бізнес-аналітик виступає ключовою ланкою між клієнтом і командою, допомагаючи приймати стратегічні рішення. Використовуючи інструменти аналізу (наприклад, Decision Tree), BA надає клієнту прозору оцінку можливостей і ризиків, щоб вибрати оптимальний шлях розвитку проєкту. Визначенні рішення демон-

струють, що роль бізнес-аналітика полягає не тільки в тому, щоб виконати запит клієнта, але й запропонувати стратегічно обґрунтовані альтернативи, які можуть оптимізувати витрати, підвищити ймовірність успіху проєкту або розширити можливості бізнесу.

Після визначення кореневого вузла, а також всіх можливих рішень можемо виконати візуалізацію першого етапу побудови дерева прийняття рішень (рис. 4.1)

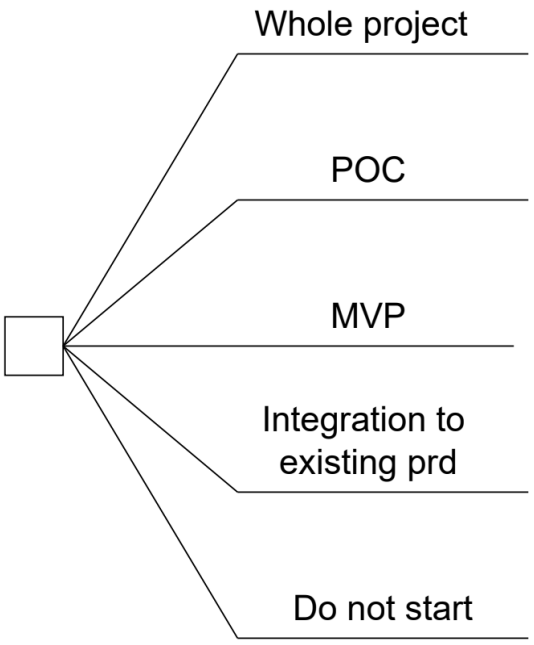

Рисунок 4.1 – Кореневий вузол

Зауважимо, що не кожен проєкт матиме усі 5 опцій. Клієнти можуть одразу окреслювати можливості для вибору, або ж сама специфіка проєкту не буде дозволяти використовувати якийсь з варіантів дерева, наприклад, інтеграцію з вже готовим рішенням. В той же час, не має сенсу для клієнтів замовляти для одного й того ж продукту одразу і PoC, і MVP. Тому тримаємо в голові, що теоретично у нас такі опції завжди є (тому ми й додаємо їх у шаблон), але на практиці деякі можливості будуть відсутні (під час створення програмного рішення, використовуємо false оператор).

Розглянемо всі альтернативні рішень як окремий корінь зі своїми гілками сценаріїв.

Почнемо з найпростішого, з останнього. У випадку відмови від проєкту ми як нічого не втрачаємо, так і нічого й не отримуємо, тобто очікувана вартість завжди буде дорівнювати нулю (рис. 4.2).

Рисунок 4.2 – Гілка варіанту «не стартуємо проєкт»

Тепер розглянемо гілку PoC + MVP. Коли клієнт замовляє виконання проєкту, перед ним може постати вибір між створенням Proof of Concept (PoC) або Minimum Viable Product (MVP). Обидва варіанти мають на меті перевірити життєздатність ідеї, однак вони істотно різняться за своїми завданнями, обсягом робіт і вартістю. Розглянемо ці аспекти більш детально, щоб зрозуміти, чому не завжди доцільно реалізовувати обидва варіанти.

Proof of Concept (PoC) є прототипом, створеним для підтвердження технічної можливості реалізації ідеї. Це базовий підхід, що демонструє, чи можна досягти бажаного функціоналу, використовуючи обрані технології та архітектуру. PoC зазвичай не призначений для кінцевих користувачів і не є продуктом, готовим до виходу на ринок. В той час, Minimum Viable Product (MVP) – це мінімально життєздатний продукт, який містить тільки основний функціонал, необхідний для задоволення ключових потреб користувачів. Головна мета MVP – отримання зворотного зв'язку від ринку, тестування попиту та визначення можливостей масштабування продукту.

Немає сенсу реалізовувати і PoC, і MVP одночасно. Поєднання PoC і MVP у рамках одного проєкту може призвести до дублювання зусиль і витрат. Основною причиною є те, що PoC підтверджує технічну здійсненність ідеї, а

MVP перевіряє ринковий попит. Зазвичай вибір залежить від стадії розвитку проєкту та запитів клієнта.

PoC вибирають, коли:

– є сумніви щодо технічної реалізовності ідеї;

– клієнт хоче протестувати нові технології, інтеграції чи архітектурні рішення;

– необхідно обґрунтувати доцільність проєкту перед інвесторами чи керівництвом.

MVP вибирають, коли:

– ідея вже має підтверджену технічну можливість реалізації;

– клієнт хоче якомога швидше вийти на ринок із мінімальним функціоналом;

– необхідно отримати зворотний зв'язок від користувачів і тестувати гіпотези.

Отже, обидва варіанти є «пробною» версією рішення продукту, але з суттєвими відмінностями з точки зору бізнесу. Для використання даних опцій у нашому дереві правильніше буде помістити їх під один альтернативний вузол (рис. 4.3), що значно спростить побудову моделі.

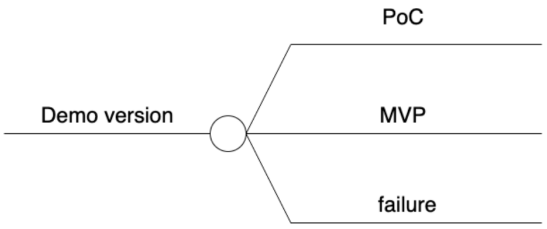

Рисунок 4.3 – Гілка варіанту «демо» версії продукту

З точки зору бізнесу і грошей є сенс розглянути питання ціни для обох варіантів. Зазвичай MVP є дорожчим варіантом, оскільки:

– вимагає більше часу та ресурсів для реалізації;

– повинен мати базовий інтерфейс, функціональність і відповідати стандартам кінцевого користувача;

– потребує залучення більшої кількості команд, таких як UX/UI-дизайнери, розробники, тестувальники.

PoC, у свою чергу, зазвичай включає лише технічний прототип і реалізується невеликою командою, що робить його значно дешевшим.

Перейдемо до опції рішення шляхом інтеграції існуючої системи (продукту). Варіант опціональний, адже не в кожному проєкті є можливість використати готові рішення. Розглянемо більш детально: уявімо ситуацію, коли клієнт замовляє інтеграцію нового рішення з вже існуючою системою, наприклад, CRM-системою чи платіжною платформою. Такий підхід є актуальним для компаній, які прагнуть розширити функціонал без необхідності створення продукту "з нуля". При цьому перед клієнтом постає вибір: інтеграція з безкоштовним open-source продуктом або використання платного сервісу чи ліцензії.

Перший варіант – інтеграція з безкоштовним open-source продуктом. Open-source рішення стають привабливим варіантом, особливо для компаній із обмеженим бюджетом. Наприклад, клієнт може інтегрувати CRM-систему з відкритим API або додати функціонал аналітики через open-source модуль. Основними перевагами такого підходу є:

– низька вартість впровадження: клієнт не витрачається на ліцензії чи підписки;

– гнучкість: код можна адаптувати під специфічні потреби бізнесу;

– широка спільнота: для популярних open-source продуктів існують форуми, документація та приклади реалізації.

Однак інтеграція з open-source продуктом має і свої виклики, а саме:

– необхідність технічної експертизи: для налаштування та кастомізації потрібні кваліфіковані розробники;

– відсутність гарантій: у разі виникнення проблем клієнт не має офіційної підтримки;

– можливі проблеми з масштабованістю: безкоштовні продукти можуть не завжди відповідати вимогам великого бізнесу.

Open-source підходить у таких випадках:

– клієнт має внутрішню команду розробників, здатних підтримувати інтеграцію;

– основний функціонал продукту відповідає потребам бізнесу;

– клієнт готовий інвестувати час у тестування та адаптацію системи.

Другий варіант – інтеграція з платним сервісом чи купівля ліцензійного продукту. Платні сервіси часто обирають компанії, які готові інвестувати в стабільне та перевірене рішення. Наприклад, клієнт може інтегрувати свою систему з хмарною платформою для аналітики чи платіжним шлюзом із гарантованою підтримкою. Основні переваги такого підходу:

– технічна підтримка: постачальники платних рішень зазвичай пропонують оперативну допомогу;

– регулярні оновлення: клієнт отримує доступ до нових функцій та поліпшень;

– швидкість впровадження: готовий функціонал дозволяє мінімізувати час на інтеграцію;

– висока надійність: постачальники гарантують стабільність та безпеку продукту.

Проте платні рішення мають і свої недоліки:

– вартість: ліцензії, підписки чи разові платежі можуть бути суттєвими;

– залежність від постачальника: якщо сервіс змінює політику або закривається, це може створити труднощі для клієнта.

Платні рішення рекомендовані, коли:

– клієнт має достатній бюджет для інвестування в якісне рішення;

– час впровадження критично важливий, і потрібне швидке розгортання;

– необхідно мінімізувати ризики, пов’язані з функціонуванням системи.

Таким чином, маємо готову гілку дерева з інтеграціями (рис. 4.4).

Остання гілка, яка залишилась – це варіант з повноцінною розробкою програмного забезпечення нашою командою. На практиці, коли клієнт замовляє розробку програмного забезпечення, і вендор (тобто наша аутсорс компанія) бере на себе виконання проєкту, в залежності від масштабу проєкту, доступних

ресурсів і технічних вимог, сейлз менеджер може запропонувати два основні підходи: повну розробку власними силами або часткову розробку з залученням сторонніх спеціалістів (outstaff чи фріланс).

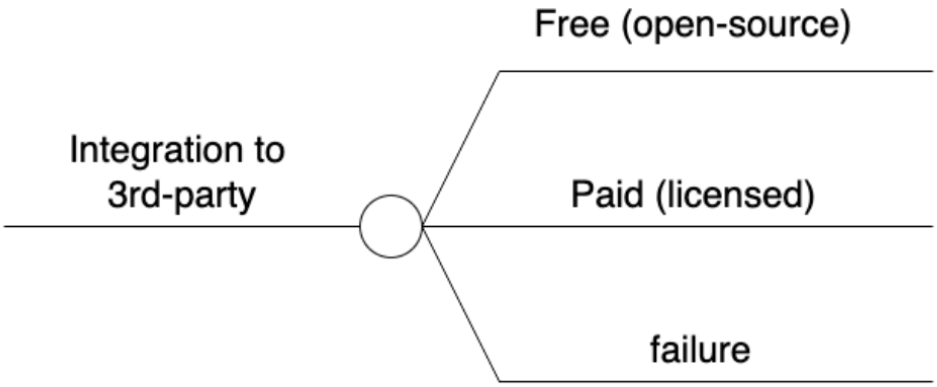

Рисунок 4.4 – Гілка варіанту інтеграції з існуючим продуктом

Розглянемо детально обидва підходи. Почнемо з повної розробки (Full Stack) силами вендора.

У цьому підході вся робота – від планування до релізу – виконується внутрішньою командою вендора. Команда включає бізнес-аналітиків, розробників, тестувальників, дизайнерів та інших спеціалістів, які забезпечують реалізацію всіх аспектів проєкту. Такий варіант зазвичай обирається для середніх і великих проєктів, де потрібен високий рівень контролю, злагоджена комунікація і чітке виконання вимог клієнта.

Переваги:

– цілісність процесу: уся команда працює в єдиній екосистемі, що забезпечує злагоджену взаємодію;

– контроль якості: вендор повністю відповідає за всі етапи розробки та може гарантувати якість продукту;

– швидкість комунікації: відсутність необхідності координуватися із зовнішніми спеціалістами скорочує час ухвалення рішень;

– безпека: чутлива інформація та код залишаються в межах компанії вендора.

Недоліки:

– вища вартість для клієнта: залучення повної внутрішньої команди завжди є дорожчим варіантом;

– обмеженість ресурсів: якщо проєкт потребує вузькопрофільної експертизи, вендору може знадобитися додатковий час на пошук і навчання спеціалістів.

Повний цикл розробки на стороні вендора найчастіше використовується для комплексних проєктів, які потребують високого рівня інтеграції між компонентами системи або спеціальних знань, які вже є у команди вендора.

Другим варіантом цього кейсу є часткова розробка (Partially In-house) із залученням зовнішніх спеціалістів. У цьому підході вендор бере на себе основну частину роботи, наприклад, розробку бекенду, архітектури системи чи девопс (сетап інфраструктури). Водночас деякі завдання (наприклад, розробка мобільних застосунків, UI/UX дизайну або тестування) передаються зовнішнім спеціалістам на аутстафф або фріланс. Цей підхід є гнучкішим і дозволяє комбінувати переваги внутрішньої команди з експертизою сторонніх фахівців.

Переваги:

– оптимізація витрат: робота з фрилансерами чи аутстафф-спеціалістами може бути дешевшою, ніж утримання великих команд у штаті;

– доступ до вузькопрофільних знань: залучення експертів дозволяє швидко вирішувати специфічні завдання без довготривалого навчання внутрішніх ресурсів;

– гнучкість: вендор може адаптувати команду залежно від етапу проєкту або змін у вимогах клієнта.

Недоліки:

– складність комунікації: координація між внутрішньою командою та зовнішніми спеціалістами може ускладнюватися через різні часові пояси чи підходи до роботи;

– потенційні ризики для безпеки: передача частини роботи стороннім фахівцям може створити ризики витоку інформації;

– розподіл відповідальності: за умови виникнення проблем може бути складно визначити, хто відповідає за їх вирішення – внутрішня команда чи зовнішній спеціаліст.

Часткова розробка часто використовується для швидких або обмежених за бюджетом проєктів, а також коли клієнту потрібна висока якість виконання вузькопрофільних завдань.

Таким чином, маємо готову гілку дерева з рішенням розробки цілого продукту (рис. 4.5).

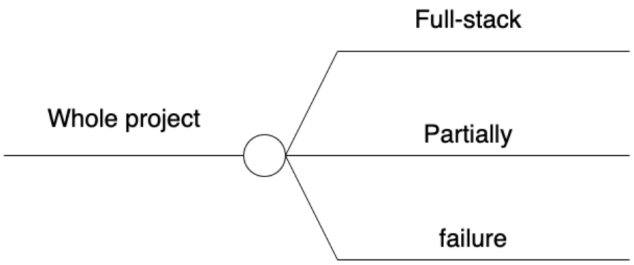

Рисунок 4.5 – Гілка варіанту цілого продукту

Попередньо на рівні проєкту ми зібрали всі можливі рішення для старту. Маємо змогу побудувати дерево повністю (рис. 4.6). На схемі бачимо 4 основні гілки рішень:

– розробка продукту повністю;

– розробка з використанням готових рішень;

– розробка PoC або MVP версії продукту;

– не робити нічого.

Перш ніж нанести змінні імовірностей, варто зазначити яким чином бізнес-аналітик або сейлз менеджер проводить їх оцінки взагалі. Кожна аутсорс компанія має свою систему оцінки таких показників, але в основі всіх лежать одні й ті ж самі принципи. Збір інформації про ймовірність старту проєкту є важливим етапом планування, особливо для вендора, який розглядає різні варіанти виконання. В основі оцінки лежать фактори репутації клієнта, ризики домену, політики компанії та специфіка ринку. В залежності від обраного підходу

ймовірності можуть суттєво варіюватися.

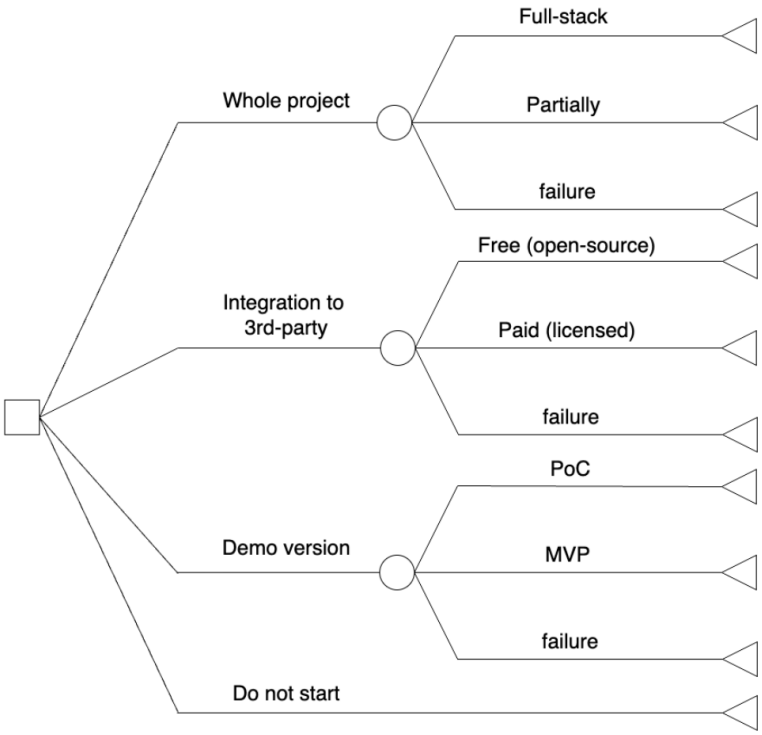

Рисунок 4.6 – Шаблон дерева рішень без змінних імовірностей

Наприклад, для розробки повного продукту, якщо проєкт реалізується через повну розробку на стороні вендора, ймовірність старту буде залежати від таких факторів:

– репутація клієнта: постійні клієнти чи компанії з високою репутацією збільшують ймовірність початку;

– складність проєкту: великі, складні проєкти потребують значних ресурсів і часу, що може знизити ймовірність старту.

Щодо вибору між Full Stack і Partially In-house:

– для Full Stack розробки ймовірність старту буде вищою, якщо клієнт очікує високого рівня контролю і не хоче залучати сторонніх спеціалістів;

– для Partially In-house ймовірність може зрости, якщо клієнт має обмежений бюджет і готовий комбінувати ресурси.

Варіанти PoC або MVP є більш ймовірними для старту, оскільки вони дозволяють клієнту мінімізувати витрати та ризики. Для PoC ймовірність старту

висока, якщо клієнт хоче перевірити технічну здійсненність ідеї. І особливо підходить для інноваційних проєктів або нових доменів. А для MVP ймовірність старту висока, якщо клієнт має обмежений бюджет, але хоче швидко протестувати ідею на ринку. Також MVP часто обирають стартапи або компанії, які планують вийти на ринок із базовим продуктом.

Щодо розробки ПЗ з використанням готових сервісів (open-source або licensed), маємо наступну картину – використання готових сервісів є популярним варіантом для швидкої інтеграції та зниження витрат. Open-source має ймовірність старту вищу, якщо клієнт прагне мінімізувати витрати і має технічні ресурси для підтримки інтеграції. Також вибір open-source залежить від специфіки домену та можливості кастомізації. В Licensed (paid) варіанті ймовірність старту зростає, якщо клієнт готовий інвестувати в надійне і перевірене рішення. Платні сервіси зазвичай обирають компанії, які цінують швидкість реалізації та стабільність роботи. Тому аналіз стекхолдерів (клієнтів) з боку аналітика тут обов’язковий.

На фінальному етапі побудови схеми звернімось до прикладу розглянутому на рис. 3.5. Та ж сама формула для $EV_{\text{вузол}}$ має місце працювати в нашому випадку для кожного вузла.

Задача буде складатися в тому, щоб внести змінні у формулу і порівняти вихідні результати для кожного вузла. В якості реалізації програми розрахунку можна використати безліч інструментів, від Excel до Phyton. Сама програма буде здатна і вирахувати найліпший варіант також. Нанесемо змінні ймовірностей, прибутків і витрат на дерево (рис 4.7).

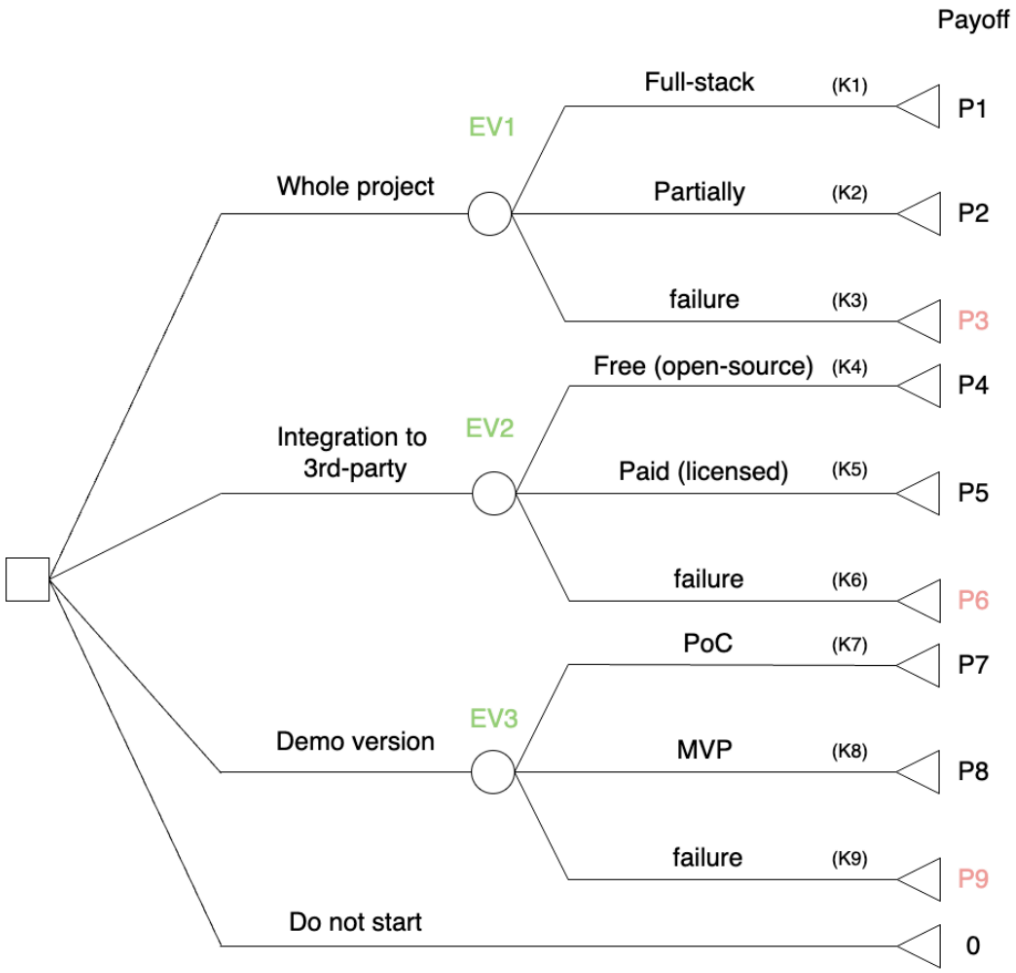

Рисунок 4.7 – Шаблон дерева рішень зі змінними імовірностей

Маємо наступні змінні:

$P_1,\ P_2,\ldots,\ P_9$ – очікуваний прибуток для кожної гілки;

$EV_1,\ EV_2,\ EV_3$ – очікувана цінність кожного рішення;

$K_1,\ K_2,\ldots,\ K_9$ – ймовірність кожного варіанта за оцінкою аналітика.

Запишемо формули для кожної гілки.

Для розрахунку цінності варіанту розробки цілого продукту:

$$EV_1 = P_1 \cdot K_1 + P_2 \cdot K_2 + P_3 \cdot K_3 .$$

Для розрахунку цінності інтеграції маємо:

$$EV_2 = P_4 \cdot K_4 + P_5 \cdot K_5 + P_6 \cdot K_6.$$

А для розрахунку цінності «пробної» версії наступна формулу:

$$EV_3 = P_7 \cdot K_7 + P_8 \cdot K_8 + P_9 \cdot K_9.$$

Після порівняння очікуваної вартості можна зробити беззаперечний вибір, оснований саме на бізнес-складовій.

4.3 Розробка і опис програми

Спочатку визначимо основну мету програми: створити систему для аналізу кількох варіантів рішень на основі введених користувачем даних, з можливістю обчислення очікуваної вигоди (Expected Value, EV) для кожного варіанту. Основні вимоги включають інтерактивний інтерфейс, валідацію введених даних, і порівняння результатів для вибору оптимального варіанту.

Далі розробляємо логіку сценаріїв – для кожного з можливих варіантів (Full Development, Demo Version, Using Existing Service) визначено структуру даних. Кожен сценарій включає кілька гілок, які відображають можливі результати реалізації (успіх/невдача) з відповідними ймовірностями, витратами та вигодами.

Приклад логіки сценаріїв:

– користувач обирає сценарії для аналізу (мінімум два);

– для кожного обраного сценарію вносить необхідні дані (імовірності, витрати, вигоди);

– забезпечується валідація (наприклад, перевірка, що ймовірності складають 1).

Визначимо структуру програми. Програма розроблена з використанням функцій для організації коду:

– функції для введення даних: запитують і валідують інформацію для кожної гілки сценарію;

– розрахункові функції: використовують формули EV для кожного сценарію;

– порівняння результатів: вибирає варіант з найвищим EV;

– інтерактивність: реалізована через input() для отримання даних від користувача.

Забезпечимо валідацію – у процесі розробки додано перевірки, наприклад:

– ймовірності для кожного сценарію мають складати 1;

– всі введені значення повинні бути числовими;

– повідомлення про помилки у разі неправильного вводу, з можливістю повторного вводу.

Після розрахунку EV для кожного сценарію програма виводить значення EV для кожного варіанту, а також рекомендує варіант з найбільшим EV.

Висновки за розділом 4

В розділі було визначено і проаналізовано основні варіанти розвитку проєкта в аутсорсових компаніях. Визначено найбільш популярні і робочі опції, які пропонують клієнтам найбільші гравці ринку України.

На основі вхідної інформації від клієнтів, їх побажань, та можливостей самої компанії, було розроблено схему шаблона дерева рішень, за допомогою якої, бізнес аналітик здатен приймати ефективні рішення для бізнесу основані на математичних розрахунках вигоди для клієнта.

За допомогою Python було побудовано базову версію програми, яка робить розрахунок найбільш цінного рішення за лічені хвилини. В подальшому програму можна кастомізовувати і робити візуально привабливішою за потреби.

5 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ ТА АНАЛІЗ МОЖЛИВИХ ЗАСТОСУВАНЬ

5.1 Обчислювальний експеримент розв’язання тестової задачі 1

Задача 1. За умовами NDA назву компаній і продуктів було видалено. Компанії необхідний веб-застосунок для використання Team Lead спеціалістами та HR-менеджерами, який за допомогою штучного інтелекту генеруватиме резюме спеціалістів компанії. Метою є покращення відсотка проданих позицій для клієнтських сервісів.

Бізнес-аналітику необхідно дослідити, проаналізувати та запропонувати оптимальний підхід до реалізації даного проєкту. Для оцінки необхідно врахувати два можливі сценарії імплементації: використання існуючих сервісів або побудова рішення своїми силами.

Після виконання аналізу ідеї маємо наступні можливі 2 сценарії.

1. Використання існуючих сервісів. Цей підхід передбачає інтеграцію з уже готовими AI-інструментами для генерації резюме. Розглядаються наступні варіанти:

а) використання безкоштовного Chat GPT Addon:

1) ймовірність успіху: 30%;

2) витрати: 3000$;

3) прибуток: 5000$;

4) особливості: мінімальні витрати, проте низька ймовірність успішного досягнення цілей через обмежений функціонал;

б) використання платного Google Gemini:

1) імовірність успіху: 60%;

2) витрати: 5000$;

3) прибуток: 10000$;

4) особливості: більша ймовірність успіху завдяки функціоналу високого рівня, але вищі витрати;

в) невдача побудови продукту:

1) ймовірність: 10%;

2) витрати: 1000$;

3) прибуток: 0$;

4) особливості: ризики пов’язані із неспроможністю адаптувати існуючі сервіси до вимог компанії.

2. Побудова проєкту своїми силами. Цей підхід передбачає розробку вебзастосунку внутрішньою командою компанії повністю.

а) успішна розробка продукту:

1) ймовірність успіху: 80%;

2) витрати: 7500$;

3) прибуток: 10000$;

4) особливості: найвищий рівень контролю та адаптації під потреби компанії, але потребує значних ресурсів;

б) невдача проєкту:

1) ймовірність: 20%;

2) витрати: 7500$;

3) прибуток: 0$;

4) особливості: ризики пов’язані з технічними складнощами та перевищенням бюджету.

Вносимо всю необхідну інформацію в таблицю значень для розрахунку прибутку (табл. 5.1).

Таблиця 5.1 Розрахунок можливого прибутку (CVapp)

	P_1, \$	P_2, \$	P_3, \$	P_4, \$	P_5, \$	P_6, \$
Можливий прибуток	2000	5000	-1000	2500	-7500	0

Далі визначимо всі ймовірності (табл. 5.2).

Таблиця 5.2 Розрахунок імовірностей кожного варіанту (CVapp)

	K_1	K_2	K_3	K_4	K_5
Імовірності варіантів	0,3	0,6	0,1	0,8	0,2

Маємо всі дані щоб побудувати дерево рішень (рис. 5.1). Використовуємо розроблений шаблон з рис. 4.7, ігноруючи гілки і підгілки які не релевантні за умовами задачі.

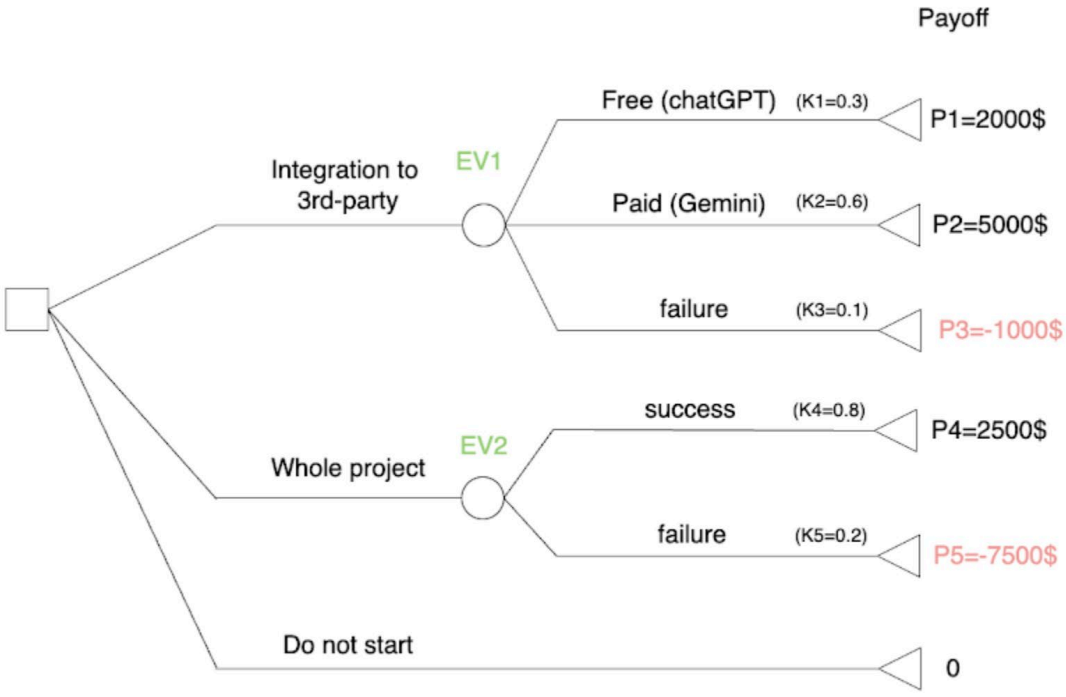

Рисунок 5.1 – Дерево рішень для розробки додатку CVapp

Залишилось порахувати цінність кожної гілки.

Для гілки використання розробки з існуючих продуктів маємо наступний розрахунок:

$$EV_1 = 2000 \cdot 0,3 + 5000 \cdot 0,6 + (-1000 \cdot 0,1) = 3500 \text{ \$}.$$

Аналогічно рахуємо цінність гілки розробки продукту повністю:

$$EV_2 = 2500 \cdot 0{,}8 + (-7500 \cdot 0{,}2) = 500 \text{ \$}.$$

Порівнюючи два значення EV_1 і EV_2, можна зробити висновок, що варіант з розробкою продукту з використанням існуючого рішення є найбільш прийнятним для нас за наявних умов. Його рекомендуємо й «продаємо» клієнту.

На рис. 5.2 наведено результат роботи програми для задачі 1.

```
Welcome to the decision-making model for business analysis!
Choose at least two branches to analyze (1 for Yes, 0 for No):
Analyze Full Development option (1 for Yes, 0 for No): 1
Analyze Demo Version option (1 for Yes, 0 for No): 0
Analyze Using Existing Service option (1 for Yes, 0 for No): 1
Full development option:
Enter probability for Full development by own team (from 0 to 1): 0.8
Enter cost for Full development by own team: 7500
Enter payoff for Full development by own team: 10000
Enter probability for Full development by third party (from 0 to 1): 0
Enter cost for Full development by third party: 0
Enter payoff for Full development by third party: 0
Enter probability for Failure of Full development (from 0 to 1): 0.2
Enter cost for Failure of Full development: 7500
Enter payoff for Failure of Full development: 0
Using existing service option:
Enter probability for Using open-source service (free) (from 0 to 1): 0.3
Enter cost for Using open-source service (free): 3000
Enter payoff for Using open-source service (free): 5000
Enter probability for Using licensed service (paid) (from 0 to 1): 0.6
Enter cost for Using licensed service (paid): 5000
Enter payoff for Using licensed service (paid): 10000
Enter probability for Failure of using existing service (from 0 to 1): 0.1
Enter cost for Failure of using existing service: 1000
Enter payoff for Failure of using existing service: 0

System recommends: Using existing service with an EV of 3500.00
```

Рисунок 5.2 – Результат роботи програми для задачі 1

5.2 Обчислювальний експеримент розв’язання тестової задачі 2

Задача 2. Клієнт хоче розробити MDM (Mobile device management) систему для керування групами пристроїв на рівні компанії (Smart office рішення). Система потребуватиме значних інвестицій в розгортанні DevOps інфраструктури на AWS, а компанія наразі не має достатньої кількості вільних спеціалістів потрібного рівня для виконання завдання. Також клієнт має дуже стислі термі-

ни виконання. Після аналізу можливих рішень маємо наступну картину варіантів.

1. Розробка PoC або MVP. Цей підхід допоможе зробити мінімальний робочий продукт або його «макет» для подальшого залучення інвесторів. Розглянемо ці два варіанта:

а) PoC:

1) ймовірність успіху: 70%;

2) витрати: 100 000$;

3) прибуток: 300 000$;

4) особливості: мінімальні витрати, з урахуванням обсягу проєкту; найбільший відсоток успіху, за рахунок простоти та створення «презентаційної» версії продукту;

б) MVP:

1) ймовірність успіху: 20%;

2) витрати: 150 000$;

3) прибуток: 350 000$;

4) особливості: мінімальний робочий продукт скоротить витрати у майбутньому, але матиме більші витрати зараз, також варіант має маленький шанс успіху бо великий ризик не вкластися в строки;

в) невдача побудови продукту:

1) ймовірність: 10%;

2) витрати: 150 000$;

3) прибуток: 0$;

4) особливості: ризики пов’язані із неспроможністю розгорнути необхідну інфраструктуру в терміни вказані замовником.

2. Побудова проєкту своїми силами. Цей підхід передбачає розробку вебзастосунку внутрішньою командою компанії повністю, або окремо знайти підрядника для виконання DevOps роботи:

а) розробка суто командою компанії:

1) ймовірність успіху: 30%;

2) витрати: 350 000$;

3) прибуток: 900 000$;

4) особливості: найвищий рівень контролю та адаптації під потреби компанії, але потребує значних ресурсів;

б) залучення зовнішньої команди DevOps:

1) ймовірність: 60%;

2) витрати: 250 000$;

3) прибуток: 900 000$;

4) особливості: додатковий пошук потрібних спеціалістів зовні;

в) невдача продукту:

1) ймовірність: 10%;

2) витрати: 350 000$;

3) прибуток: 0$.

Маємо всю необхідну інформацію для створення таблиці значень. Першою заповнимо таблицю прибутку (табл. 5.3).

Таблиця 5.3 Розрахунок можливого прибутку (MDM)

	P_1, $	P_2, $	P_3, $	P_4, $	P_5, $	P_6, $	P_7, $
Можливий прибуток	200000	200000	-150000	550000	650000	-350000	0

Далі визначимо всі ймовірності (табл. 5.4).

Таблиця 5.4 Розрахунок імовірностей кожного варіанту (MDM)

	K_1	K_2	K_3	K_4	K_5	K_6
Імовірності варіантів	0,7	0,2	0,1	0,3	0,6	0,1

Будуємо дерево рішень використовуючи данні з таблиць вище (рис. 5.3).

Розраховуємо цінність кожного варіанту аналогічно попереднього прик-

ладу, використовуючи шаблон.

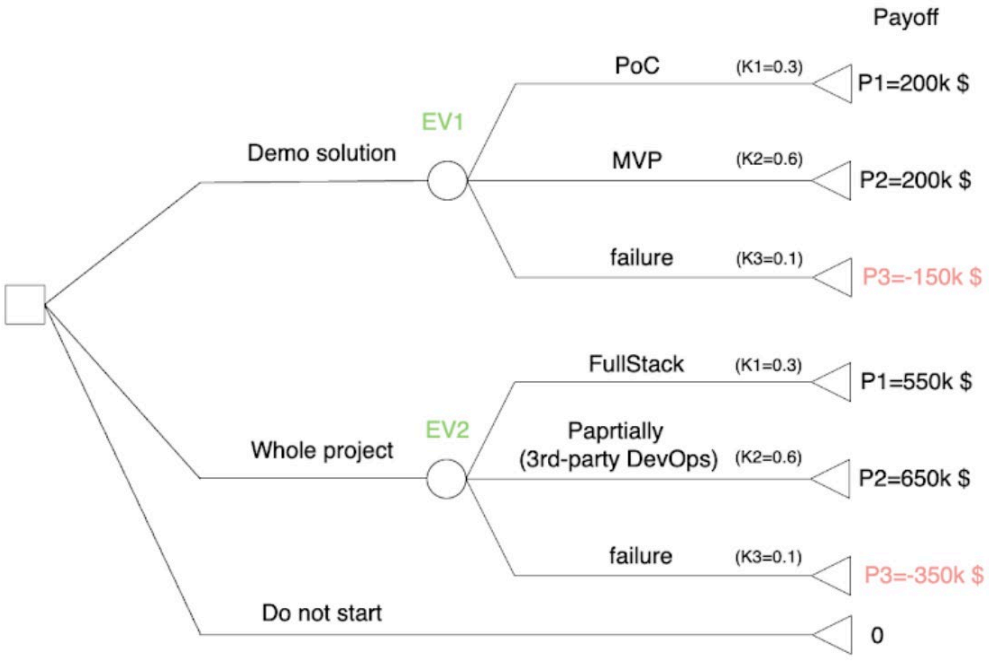

Рисунок 5.3 – Дерево рішень для розробки MDM

Для пробної версії продукту маємо наступну формулу:

$$EV_1 = 200000 \cdot 0,3 + 200000 \cdot 0,6 + (-150000 \cdot 0,1) = 165000\$.$$

Аналогічно рахуємо цінність гілки розробки продукту повністю:

$$EV_2 = 550000 \cdot 0,3 + 650000 \cdot 0,6 + (-350000 \cdot 0,1) = 520000\$.$$

Таким чином, найліпшим варіантом для нас буде гілка з розробкою проєкту повністю.

Як бачимо розроблений шаблон працює і дозволяє оптимізувати процес прийняття рішень.

На рис. 5.4 наведено результат роботи програми для задачі 2.

Як бачимо, маємо такий самий аутпут і від розробленої програми.

```
Welcome to the decision-making model for business analysis!
Choose at least two branches to analyze (1 for Yes, 0 for No):
Analyze Full Development option (1 for Yes, 0 for No): 1
Analyze Demo Version option (1 for Yes, 0 for No): 1
Analyze Using Existing Service option (1 for Yes, 0 for No): 0
Full development option:
Enter probability for Full development by own team (from 0 to 1): 0.3
Enter cost for Full development by own team: 350000
Enter payoff for Full development by own team: 900000
Enter probability for Full development by third party (from 0 to 1): 0.6
Enter cost for Full development by third party: 250000
Enter payoff for Full development by third party: 900000
Enter probability for Failure of Full development (from 0 to 1): 0.1
Enter cost for Failure of Full development: 350000
Enter payoff for Failure of Full development: 0
Demo version option:
Enter probability for Proof of Concept (PoC) (from 0 to 1): 0.7
Enter cost for Proof of Concept (PoC): 100000
Enter payoff for Proof of Concept (PoC): 300000
Enter probability for Minimum Viable Product (MVP) (from 0 to 1): 0.2
Enter cost for Minimum Viable Product (MVP): 150000
Enter payoff for Minimum Viable Product (MVP): 350000
Enter probability for Failure of Demo version (from 0 to 1): 0.1
Enter cost for Failure of Demo version: 150000
Enter payoff for Failure of Demo version: 0

System recommends: Full development with an EV of 520000.00
```

Рисунок 5.4 – Результат роботи програми для задачі 2

Висновки за розділом 5

В розділі 5 експериментальним шляхом було протестовано розроблену систему з обох боків – і з точки зору мануального розрахунку, і з точки зору використання розробленого програмного рішення.

Система впоралась з поставленими задачами і довела свою ефективність. За допомогою програми, аналіз і математичний розрахунок цінностей варіантів, а також саме прийняття рішення займає не більше 5 хвилин, при найбільш розгалужених варіантах постановки задачі. У подальшому систему можна масштабувати або кастомізувати під більш вузькі задачі на рівні ініціативи або фічі в уже існуючих проєктах.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було успішно досягнуто поставленої мети – розробка автоматизованої системи прийняття рішень для декількох проєктів з можливістю її масштабування на рівні компанії. У процесі роботи було виконано аналіз життєвого циклу розробки програмного забезпечення в аутсорсових компаніях, визначено ключові точки прийняття рішень та досліджено методи їх автоматизації.

Проведений системний аналіз дозволив сформулювати основні вимоги до системи підтримки рішень. Було обрано техніку моделювання рішень (Decision Tree) як найбільш відповідну для структуризації процесів ухвалення рішень у бізнес-аналізі. Результати дослідження підтвердили ефективність цієї техніки для створення прозорих і масштабованих процесів ухвалення рішень.

У роботі розглянуто широкий спектр інструментів для побудови дерев рішень, таких як Microsoft Excel, Lucidchart, Python (sklearn), Draw.io та інші. Проведено їх аналіз, визначено переваги та недоліки, а також рекомендовано оптимальні інструменти залежно від складності завдань і ресурсів компанії.

На основі реальних кейсів було побудовано приклади дерев рішень для різних сценаріїв: розробки продукту, використання готових сервісів або інтеграції з існуючими системами. Це дало змогу оцінити економічну доцільність і ризики кожного варіанту, а також забезпечило можливість автоматизованого розрахунку очікуваної вартості альтернатив.

Особливу увагу було приділено pre-sale/discovery фазі, на якій бізнес-аналітик виконує збір вимог, аналіз ринку та формулювання пропозицій. Було розроблено шаблон для автоматизації ухвалення рішень на цьому етапі, що дозволяє значно скоротити час і зусилля, необхідні для оцінки проєктів.

Таким чином, практичним шляхом було доведено, що результати роботи можуть бути використані у реальному житті для оптимізації системи прийняття рішень на рівні менеджменту проєкту, що робить результат роботи актуальним для ІТ-компаній, які працюють у сфері аутсорсу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. BABOK® Guide. Business Analysis Body of Knowledge. Kyiv : International Institute of Business Analysis, 2015. 500 p.

2. Dumas M., La Rosa M., Mendling J., Reijers H. A. Fundamentals of Business Process Management. 2nd ed. Berlin : Springer, 2018. 527 p.

3. Otroshchenko A. Business analysis and modelling of decision-making processes in project management. *Learning & Teaching: after War and during Peace [Electronic Edition]: Conference Proceedings of III International Scientific & Practical Conference*, Kharkiv, Ukraine, 8 November, 2024. Kharkiv : KNPU, 2024. P. 236.

4. Harmon P. Business Process Change: A Guide for Business Managers and BPM and Six Sigma Professionals. 3rd ed. Burlington : Morgan Kaufmann, 2014. 520 p.

5. Maciaszek L. A. Requirements Analysis and System Design: Developing Information Systems with UML. 3rd ed. Harlow : Pearson Education, 2007. 614 p.

6. Debevoise T., Geneva R. The Microguide to Process Modeling in BPMN 2.0. 3rd ed. United States : CreateSpace Independent Publishing Platform, 2014. 236 p.

7. Аблєєва І. Ю., Пляцук Л. Д. SWOT-аналіз соціо-економіко-екологічних систем. Кривий Ріг : Видавничий дім, 2016. 158 с.

8. Art of BA. URL: https://www.artofba.com/ (дата звернення: 18.10.2024).

9. International Institute of Business Analysis (IIBA®). URL: https://www.iiba.org/ (дата звернення: 12.10.2024).

10. Business Analyst Mentor. URL: https://businessanalystmentor.com/ (дата звернення: 19.10.2024).

11. Rouse, W. B. Enterprise Transformation: Understanding and Enabling Fundamental Change. Hoboken: John Wiley & Sons, 2006. 400 p.

12. Project Management Institute. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). 6th ed. Newtown Square, PA: Project Management Institute, 2017. 756 p.

13. IBM. URL: https://www.ibm.com/ (дата звернення: 15.10.2024).

14. M3. URL: https://pm3us.com/knowledge-hub/decision-making-models/ (дата звернення: 12.12.2024).

15. GeeksforGeeks. Applications of Python in Artificial Intelligence and Decision-Making Systems. URL: https://www.geeksforgeeks.org/applications-of-python (дата звернення: 30.12.2024).

16. Python Software Foundation. Python: A Programming Language that Empowers Developers Worldwide. URL: https://www.python.org (дата звернення: 30.12.2024).

17. GitHub. URL: https://chenweixiang.github.io/docs/Decision_Trees.pdf/ (дата звернення: 12.12.2024).