МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 124 «Системний аналіз»
Освітня програма «Інформаційні технології інтелектуального аналізу даних та проєктування програмних систем»

«Допущений до захисту»
Завідувач кафедри
проф. Приходько С.Б.
«____»_____2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «магістр»

на тему: Математична модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java та програма для її реалізації

Винонала: студентка групи 6153м
Малюк А.С.
Керівник роботи:
доцент кафедри ПЗАС, к.т.н
Макарова Л.М.

Миколаїв-2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 124 «Системний аналіз»
Освітня програма «»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Латанська Л.О.
(підпис)
« 15 » жовтня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «магістр»

Студенту Малюк Ананстасії Сергівні
(Прізвище, ім'я, по батькові)

1. Тема роботи: Математична модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java та програма для її реалізації
Керівник роботи Макарова Л. М., доц. каф. ПЗАС, к.т.н., доц.
Затверджені наказом ректора № 1222-УЧ від « 15 » 10 2025 р.
2. Термін подання роботи: 08.12.2025 р.
3. Вихідні дані по роботі: _____

4. Перелік питань, що належать до розробки (найменування розділів) _____
Вступ (Актуальність теми. Зв'язок роботи з науковими програмами, планами, темами. Мета і завдання дослідження. Об'єкт дослідження. Предмет дослідження. Методи дослідження. Наукова новизна одержаних результатів. Практичне значення одержаних результатів. Особистий внесок здобувача. Апробація результатів досліджень**.** Публікації.); Огляд літератури та обґрунтування необхідності проведення досліджень за обраною темою; Викладення результатів власних досліджень з висвітленням того нового, що пропонується; Розділ з розробки програмного забезпечення; Організаційно-економічний розділ; Розділи з охорони праці та охорони навколишнього середовища; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення)

5. Перелік презентаційних матеріалів: Актуальність теми. Мета і завдання дослідження. Об'єкт дослідження. Предмет дослідження. Методи дослідження. Наукова новизна одержаних результатів. Практичне значення одержаних результатів. Апробація результатів досліджень. Публікації. Існуючі регресійні моделі для оцінювання метрики RFC. Нелінійна регресійна модель для оцінювання метрики RFC застосунків з відкритим кодом на Java. Постановка задачі на розробку проекту програми, Загальносистемні рішення ПС, Інформаційне забезпечення ПС, Програмне забезпечення ПС, Результати випробувань ПС.
Висновки _____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 27.10.2025 р. _____

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу ВСТУП	28.10.2025	ДП*
2.	Підготовка розділу з огляду літератури та обґрунтування необхідності проведення досліджень за обраною темою	30.10.2025	ДП*
3.	Підготовка розділу (ів) за результатами власних досліджень	01.11.2025	ДП*
4.	Підготовка розділу з розробки програмного забезпечення	26.11.2025	
5.	Підготовка організаційно-економічного розділу	28.11.2025	
6.	Підготовка розділу з охорони праці	02.12.2025	
7.	Підготовка розділу з охорони навколишнього середовища	03.12.2025	
8.	Підготовка розділу ВИСНОВКИ	04.12.2025	
9.	Оформлення списку використаних джерел та додатків	05.12.2025	
10.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версії роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	08.12.2025	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку)
11.	Підготовка презентації та доповіді	12.12.2025	
12.	Попередній захист роботи на засіданні кафедри ПЗАС	15.12.2025	
13.	Подання на кафедру ПЗАС електронних версії наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді	19.12.2025	

* - за результатами (розділ звіту) дослідницької практики (ДП), яка була з 01.09.2025 по 26.10.2025 р.

Студент _____ Малюк А.С.
(підпис) (ПІБ)

Керівник роботи _____ Макарова Л.М.
(підпис) (ПІБ)

РЕФЕРАТ

Малюк Анастасія Сергіївна

«Математична модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java та програма для її реалізації»

Кваліфікаційна робота на здобуття освітнього рівня магістра зі спеціальності 124 - «Системний аналіз». Національний університет кораблебудування імені адмірала Макарова, 2025 р.

Обсяг роботи: 117 стор., 6 табл., 16 рис., 23 використаних джерела, 5 додатків.

Актуальність теми: Розроблення математичної моделі для автоматизації прийняття рішень у сфері e-commerce забезпечить оптимізацію процесів оцінювання якості, підвищення ефективності розробки та вдосконалення програмних систем e-commerce на Java.

Мета та завдання дослідження. Метою роботи є підвищення достовірності прийняття рішень щодо якості програмних систем з відкритим кодом, що розробляються мовою Java у сфері e-commerce.

Об'єктом дослідження є процес оцінювання якості програмних систем e-commerce на Java.

Предмет дослідження є математичні моделі для прийняття рішень щодо якості програмних систем e-commerce на Java, що використовують метрики RFC, CBO, WMC.

Методи дослідження: математичного моделювання, теорії ймовірності, математичної статистики та регресійного аналізу, об`єктно-орієнтованого програмування.

Наукова новизна одержаних результатів. Удосконалено математичну модель для прийняття рішень щодо якості програмних систем e-commerce на Java із застосуванням метрик RFC, CBO та WMC шляхом використання нормалізуючого перетворення у формі десяткового логарифму з урахуванням викидів регресії. Запропонована модель, порівняно з існуючими моделями, враховує інтервали значень метрик програмних систем e-commerce на Java, що забезпечує підвищення достовірності оцінювання якості відповідних програмних систем.

Практичне значення одержаних результатів. Розроблено програму, яка реалізує побудовану математичну модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java.

Апробація результатів дослідження. Основні положення та результати досліджень, викладені у роботі, були оприлюднені на VIII Всеукраїнській науково-практичній інтернет-конференції студентів, аспірантів та молодих вчених за тематикою «Сучасні комп'ютерні системи та мережі в управлінні» (24 листопада 2025 р., м. Херсон, м. Хмельницький).

Публікація. За результатами досліджень опубліковано одну наукову працю, а саме матеріали конференції.

Ключові слова: регресійна модель, оцінювання якості, e-commerce, метрики RFC, CBO, WMC, застосунок з відкритим кодом, Java.

ABSTRACT

Maliuk Anastasia Serhiivna

"Mathematical model for automating decision-making regarding the quality of e-commerce software systems in Java and a software for its implementation"

Qualification work for obtaining a master's degree in speciality 124 - ‘System Analysis’. Admiral Makarov National University of Shipbuilding, Mykolayiv 2025.

The qualification work are 117 pages, 6 tables, 16 figures, 23 references, 5 appendices.

Relevance of the topic of the work: Developing a mathematical model for automating decision-making in e-commerce will optimise quality assessment processes, improve the efficiency of development, and enhance Java-based e-commerce software systems.

The purpose and objectives of the study: The purpose of this work is to improve the reliability of decision-making regarding the quality of open source software systems developed in Java in the field of e-commerce.

The object of the study is the process of evaluating the quality of Java-based e-commerce software systems.

The subject of the study is a Mathematical models for decision-making regarding the quality of Java-based e-commerce software systems using RFC, CBO, and WMC metrics.

The methods of the study: mathematical modelling, probability theory, mathematical statistics and regression analysis, object-oriented programming.

The scientific novelty of obtained results: A mathematical model for making decisions on the quality of Java-based e-commerce software systems using RFC, CBO, and WMC metrics has been improved by applying a normalising transformation in the form of a decimal logarithm, taking into account regression outliers. Compared to existing models, the proposed model takes into account the intervals of values of metrics for Java-based e-commerce software systems, which improves the reliability of assessing the quality of the corresponding software systems.

The practical significance of obtained results: A programme has been developed that implements a mathematical model for automating decision-making regarding the quality of Java-based e-commerce software systems.

Approbation of study results: The main provisions and results of the research presented in this work were published at the VIII All-Ukrainian Scientific and Practical Internet Conference.

Publication: Based on the research results, one scientific work was published, namely the conference materials.

Keywords: regression model, quality assessment, e-commerce, RFC, CBO, WMC metrics, open source application, Java.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДНИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ 8

ВСТУП 9

1 АНАЛІЗ ІСНУЮЧИХ МОДЕЛЕЙ ДЛЯ ОЦІНЮВАННЯ МЕТРИКИ RFC ЗАСТОСУНКІВ НА JAVA 13

1.1 Особливості використання мови Java для розробки e-commerce програмних систем 13

1.2 Аналіз існуючих моделей для оцінювання метрик RFC застосунків на Java 16

1.3 Обґрунтування необхідності проведення досліджень за обраною темою 18

1.4 Висновки за розділом 1 20

2 МАТЕМАТИЧНА МОДЕЛЬ ДЛЯ АВТОМАТИЗАЦІЇ ПРИЙНЯТТЯ РІШЕНЬ ЩОДО ЯКОСТІ ПРОГРАМНИХ СИСТЕМ E-COMMERCE НА JAVA 22

2.1 Аналіз наявних математичних моделей оцінювання якості програмних систем та вибір факторів для автоматизації прийняття рішень щодо якості e-commerce програмних систем на Java 22

2.2 Попередня обробка та аналіз емпіричних даних для побудови математичної моделі оцінювання якості програмних систем e-commerce, реалізованих мовою Java 26

2.3 Побудова математичної моделі для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java 29

2.4 Висновки за розділом 2 34

3 РОЗРОБКА ПРОЄКТНИХ РІШЕНЬ ДЛЯ ПРОГРАМНОЇ СИСТЕМИ АВТОМАТИЗАЦІЇ ПРИЙНЯТТЯ РІШЕНЬ ЩОДО ЯКОСТІ ПРОГРАМНИХ СИСТЕМ E-COMMERCE, РЕАЛІЗОВАНИХ МОВОЮ JAVA 36

3.1 Загальносистемні проєктні рішення для програмної системи оцінювання якості e-commerce ПС на Java 36

3.1.1 Вибір засобів та нотацій моделювання для програмної системи оцінювання якості 37

3.1.2 Розробка діаграми варіантів використання програмної системи автоматизації прийняття рішень щодо якості 38
3.1.3 Специфікації варіантів використання програмної системи оцінювання якості e-commerce систем..... 41
3.1.4 Сценарії основних варіантів використання програмної системи оцінювання якості 43
3.2 Проєктні рішення з інформаційного забезпечення програмної системи оцінювання якості 45
3.2.1 Концептуальна модель даних програмної системи оцінювання якості e-commerce ПС 46
3.2.2 Логічна модель даних програмної системи оцінювання якості..... 47
3.3 Проєктні рішення з програмного забезпечення програмної системи оцінювання якості 49
3.3.1 Вибір мови програмування та платформи для реалізації програмної системи оцінювання якості 50
3.3.2 Фізична модель даних програмної системи оцінювання якості 51
3.3.3 Модель класів програмної системи автоматизації прийняття рішень щодо якості 53
3.3.4 Моделі взаємодії компонентів програмної системи оцінювання якості 54
3.3.5 Тестування програмної системи оцінювання якості e-commerce ПС.. 56
3.3.6 Випробування програмної системи автоматизації прийняття рішень щодо якості 57
3.4 Висновки за розділом 3 60
4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОЇ СИСТЕМИ ДЛЯ ОЦІНЮВАННЯ ЯКОСТІ E-COMMERCE СИСТЕМ НА JAVA..... 62
4.1 Розрахунок витрат на створення та впровадження програмної системи62
4.2 Розрахунок економічної ефективності розробки та впровадження програмної системи 65
Висновки за розділом 4 68
5 ОХОРОНА ПРАЦІ..... 71
5.1 Аналіз шкідливих та небезпечних факторів у офісі комп’ютерної фірми 72

5.2 Розрахунок системи пожежогасіння у офісі комп’ютерної фірми..... 73
5.3 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів..... 75
6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА..... 78
6.1 Вступ..... 78
6.2 Аналіз існуючої проблеми необхідності охорони навколишнього середовища..... 79
6.3 Забруднення навколишнього середовища співробітниками офісу комп’ютерної фірми..... 80
6.4 Розробка заходів щодо зменшення забруднення..... 81
ВИСНОВКИ..... 83
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ..... 85
ДОДАТОК А – Технічне завдання на розробку програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java..... 88
ДОДАТОК Б – Код програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java..... 92
ДОДАТОК В – Опис програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java..... 108
ДОДАТОК Г – Інструкція користувача програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java..... 110
ДОДАТОК Д – Програма та методика випробувань програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java..... 112

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДНИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення

ПС - програмна система;

CBO – Coupling Between Object classes

JVM – Java Virtual Machine

MMRE - a mean magnitude of relative error;

PRED - percentage of prediction;

RFC – Response For a Class

WMC – Weighted Methods per Class

ВСТУП

Актуальність теми. Одним із ключових завдань сучасної розробки є визначення якості програмних систем. Відомо, що метрики об’єктно-орієнтованого дизайну RFC, CBO та WMC широко застосовуються для оцінювання якості програмних систем, зокрема тих, що розробляються мовою Java [1–3]. У роботах [1, 2] показано, що для низки відкритих Java-застосунків значення цих метрик на рівні програми можуть використовуватися для визначення її загальної якості. Зокрема, у [1] встановлено, що програмна система характеризується високою якістю, якщо середні значення її метрик RFC, CBO та WMC належать до відповідних довірчих інтервалів, отриманих на основі статистичної моделі застосунків, розроблених на Java. Однак інтервальні оцінки для цілісної характеристики якості ПЗ на рівні застосунку досліджені поки що недостатньо.

Звісно, метрика RFC є статистично залежною від метрик CBO та WMC [2, 3]. У роботі [2] побудовано нелінійну регресійну модель для оцінювання середнього значення RFC в залежності від показників CBO та WMC. Модель одержано шляхом застосування багатовимірних нормалізуючих перетворень та подальшого проведення лінійної регресії у нормалізованому просторі з наступним зворотним перетворенням. Такий підхід дозволяє отримати нелінійну регресійну залежність вигляду яка узгоджується з реальним розподілом метрик Java-застосунків.

Важливо зазначити, що використання лінійних моделей не завжди є коректним. Наприклад, у [4] показано, що розподіли залишків для регресійних моделей метрик об’єктно-орієнтованого дизайну часто не є гаусовими, що порушує базові припущення лінійної регресії. У таких випадках доцільним є застосування саме нелінійних регресійних моделей, побудованих на основі нормалізованих даних [2–4].

Згадані моделі є універсальними для програмних систем, що розробляються мовою Java, тому їх можна застосовувати і для оцінювання якості програмних систем e-commerce, однак у системах цього типу показники RFC, CBO та WMC є особливо важливими, оскільки відображають структурну складність, зв’язність компонентів та обсяг функціональності, що безпосередньо впливає на масштабованість, підтримуваність та надійність e-commerce платформ, тому доречність застосування розглянутих моделей потребує подальшого підтвердження. Отже, подальше використання нелінійних регресійних моделей для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java є обґрунтованим і підтверджує актуальність теми дослідження.

Мета і завдання дослідження. Метою роботи є підвищення достовірності прийняття рішень щодо якості програмних систем з відкритим кодом, що розробляються мовою Java у сфері e-commerce.

Для досягнення поставленої мети потрібно вирішити такі завдання.

1. Проаналізувати існуючі регресійні моделі щодо якості програмних систем, зокрема веб-систем e-commerce.

2. Здійснити підбір відповідної сукупності даних для e-commerce застосунків з відкритим кодом на Java, на основі яких буде будуватися математична модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java, та перевірити вибірку на наявність багатовимірних викидів.

3. Удосконалити нелінійну регресійну модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java, використовуючи метрики RFC, CBO та WMC шляхом застосування нормалізуючого перетворення.

4. На основі запропонованої нелінійної регресійної моделі розробити програму для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java.

Об'єктом дослідження є процес оцінювання якості програмних систем e-commerce на Java.

Предметом дослідження є математичні моделі для прийняття рішень щодо якості програмних систем e-commerce на Java, що використовують метрики RFC, CBO, WMC.

Методи дослідження. У вирішенні поставлених завдань використано методи теорії ймовірності, математичної статистики, регресійного аналізу, виявлення багатовимірних викидів, а також нормалізуючі перетворення для побудови коректних нелінійних регресійних моделей.

Наукова новизна. Удосконалено математичну модель для прийняття рішень щодо якості програмних систем e-commerce на Java із застосуванням метрик RFC, CBO та WMC шляхом використання нормалізуючого перетворення у формі десяткового логарифму з урахуванням викидів регресії. Запропонована модель, порівняно з існуючими моделями, враховує інтервали значень метрик програмних систем e-commerce на Java, що забезпечує підвищення достовірності оцінювання якості відповідних програмних систем.

Отримані результати є важливими для e-commerce платформ на Java, де складність компонентів, рівень зв'язності та кількість відповідей класів безпосередньо впливають на масштабованість і надійність функціонування програмних систем.

Практичне значення одержаних результатів. Розроблено програму для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java. Запропонований інструмент може бути використаний для підвищення достовірності оцінювання якості архітектури та структурної складності веб-систем e-commerce.

Особистий внесок здобувача. Кваліфікаційна робота є самостійно виконаною працею. Усі результати, подані в роботі, отримані автором особисто. У роботі [5] здобувачеві належать збір і аналіз емпіричних даних

метрик e-commerce програмних систем на Java, а також попередня обробка даних для побудови нелінійної регресійної моделі оцінювання якості.

Апробація результатів досліджень. Основні положення та результати досліджень, викладені у роботі, були оприлюднені на VIII Всеукраїнській науково-практичній інтернет-конференції студентів, аспірантів та молодих вчених за тематикою «Сучасні комп’ютерні системи та мережі в управлінні» (24 листопада 2025 р., м. Херсон, м. Хмельницький).

Публікації. За результатами досліджень опубліковано одну наукову працю, а саме матеріали конференції.

1 АНАЛІЗ ІСНУЮЧИХ МОДЕЛЕЙ ДЛЯ ОЦІНЮВАННЯ МЕТРИКИ RFC ЗАСТОСУНКІВ НА JAVA

1.1 Особливості використання мови Java для розробки e-commerce програмних систем

Мова програмування Java залишається однією з ключових технологій у сучасній індустрії розроблення програмного забезпечення. Її популярність зумовлена поєднанням зрілості екосистеми, технологічної стабільності та широкого спектра промислових рішень, що спираються саме на цю платформу. За даними міжнародних рейтингів, зокрема TIOBE Index, Java протягом багатьох років зберігає високі позиції у переліку найпоширеніших мов програмування [6]. Це свідчить про її стійкі конкурентні переваги та масштабне використання у різних галузях цифрової економіки – від фінансових сервісів і телекомунікацій до систем електронної комерції та державних інформаційних платформ. Одним із факторів, що забезпечує таку універсальність Java, є її платформна незалежність: принцип write once, run anywhere дає можливість запускати програмні продукти на будь-якому середовищі, де працює JVM

У контексті e-commerce Java набула особливого значення, оскільки цей сегмент розроблення програмного забезпечення висуває підвищені вимоги до продуктивності, масштабованості, безпеки та безперервної доступності систем. Електронні торговельні платформи часто працюють у режимі 24/7 та обслуговують тисячі або навіть мільйони користувачів одночасно. Тому технологічні рішення мають забезпечувати низький час відгуку, стійкість до пікових навантажень і коректну обробку великої кількості паралельних транзакцій. У цьому контексті Java показує себе як технологія, що здатна ефективно працювати в умовах високих навантажень завдяки багаторівневій

архітектурі, розвиненим механізмам керування потоками та оптимізованій роботі JVM [7].

Важливим аспектом використання Java для e-commerce є доступність комплексної екосистеми фреймворків, яка значною мірою спрощує реалізацію бізнес-логіки, інтеграцію з зовнішніми сервісами та підтримку розподіленої інфраструктури. Такі платформи, як Spring Framework, Spring Boot, Spring Cloud, Java EE та Jakarta EE надають стандартизовані механізми для роботи з базами даних, реалізації транзакційних процесів, розроблення веб-сервісів та побудови мікросервісних архітектур [7, 8]. Для e-commerce застосунків це передбачає створення модульної архітектури, яка забезпечує чітку структурованість, можливість масштабування та розширення, а також інтеграцію з платіжними системами, логістичними сервісами й CRM-платформами. Значна кількість промислових рішень у сфері e-commerce, серед яких SAP Hybris, Broadleaf Commerce та IBM WebSphere Commerce, реалізовані на основі Java. Це підтверджує її ефективність і придатність для побудови складних високонавантажених систем.

Не менш суттєвою перевагою є механізми забезпечення безпеки. Системи електронної комерції працюють із конфіденційними даними: платіжною інформацією, персональними даними клієнтів, деталями замовлень тощо. Java забезпечує широкий спектр засобів криптографії, аутентифікації та авторизації, включаючи вбудовані модулі Java Security API, підтримку OAuth 2.0, JWT, TLS/SSL та інші механізми. Завдяки цьому розробники можуть створювати рішення, які відповідають міжнародним стандартам безпеки та захищають e-commerce інфраструктуру від типових сучасних загроз.

Окрему увагу варто приділити інструментам масштабування Java-застосунків. Використання JVM дає змогу оптимізувати виконання коду під конкретну апаратну конфігурацію, що особливо важливо для систем, які працюють у хмарних середовищах та контейнеризованих інфраструктурах. Інтеграція з Docker, Kubernetes та сервісами провідних хмарних платформ

(Amazon Web Services, Google Cloud Platform, Microsoft Azure) робить Java ефективним інструментом для побудови сучасної високодоступної інфраструктури e-commerce рішень [9]. Підтримка мікросервісної архітектури, орієнтованої на автономність модулів і їхню незалежну масштабованість, суттєво підвищує стійкість таких систем до відмов та забезпечує гнучкість у їхньому розвитку.

Окремою перевагою Java є активна спільнота та широкий набір інструментів, спрямованих на автоматизацію тестування, оцінювання якості та аналітику програмного коду. Використання фреймворків JUnit, Mockito, JaCoCo, SonarQube, а також застосування розвинених метрик складності та якості коду забезпечує високий рівень програмної інженерії та відповідність систем вимогам надійності [10]. Для e-commerce платформ, де важливо мінімізувати ризик помилок і оптимізувати бізнес-процеси, це має особливе значення. Крім того, здатність Java-застосунків функціонувати в умовах складної логіки та великої кількості компонентів робить Java оптимальною основою для проєктів, у яких ключову роль відіграє оцінювання якості програмного забезпечення за метриками рівня класів (наприклад, RFC, CBO, WMC).

Таким чином, застосування Java у розробці e-commerce програмних систем забезпечує комплекс переваг, що охоплюють архітектурні, технологічні, інфраструктурні та якісні аспекти. Поєднання високої продуктивності, можливостей масштабування, широкої підтримки бізнес-орієнтованих фреймворків, потужних механізмів безпеки та стандартів індустріальної якості коду робить Java однією з найефективніших платформ для створення сучасних систем електронної комерції. Завдяки цьому Java зберігає позиції провідної технології у сферах, де необхідні стабільність, надійність та здатність працювати у високонавантажених середовищах, що є ключовими характеристиками e-commerce сегмента.

1.2 Аналіз існуючих моделей для оцінювання метрик RFC застосунків на Java

Одним із ключових завдань сучасної розробки є визначення якості програмних систем. У практиці оцінювання якості ПЗ широко застосовуються метрики об’єктно-орієнтованого дизайну, серед яких важливе місце займають RFC, CBO та WMC [1–3]. У дослідженнях [1, 2] наведено приклади використання середніх та інтервальних оцінок цих показників для визначення рівня якості Java-застосунків. Зокрема, у роботі [1] запропоновано метод формування довірчих і прогнозних інтервалів для метрики RFC, що дозволяє визначати якість ПЗ на основі статистичних характеристик множини Java-проєктів.

Разом із тим питання побудови надійних інтервальних оцінок на рівні програмної системи досліджене неповною мірою. У роботі [1] авторами побудовано рівняння лінійної регресії для оцінювання середнього значення RFC у залежності від метрик CBO та WMC:

$$\hat{Y} = \hat{b}_0 + \hat{b}_1 X_1 + \hat{b}_2 X_2 \qquad (1.1)$$

де: $\hat{Y}$ позначає прогнозоване значення RFC;

X_1 і X_2 – значення метрик CBO та WMC відповідно;

$\hat{b_0}$, $\hat{b_1}$, $\hat{b_2}$ – параметри регресійної моделі.

Попри те, що лінійні регресійні моделі мають широке застосування, їх використання не завжди можна вважати виправданим з позицій статистичної науки. Коли розподіл похибок моделі суттєво відхиляється від нормального, що часто трапляється під час аналізу програмних метрик, точність таких моделей значно погіршується [3]. За таких умов більш результативними виявляються нелінійні методи регресії, які дають змогу врахувати реальні характеристики даних та складну структуру взаємозалежностей між показниками [11, 12].

У найпростішому випадку залежність RFC від CBO та WMC описували за допомогою лінійної регресії виду:

$$Y = \hat{b}_0 + \hat{b}_1 X_1 + \hat{b}_2 X_2 + \varepsilon \quad (1.2)$$

де: X_1 та X_2 – значення метрик CBO і WMC відповідно;

ε – випадкова величина, що має відповідати нормальному розподілу;

$\hat{b}_0$, $\hat{b}_1$, $\hat{b}_2$ – параметри регресійної моделі.

Процедуру побудови нелінійної регресійної моделі для оцінювання метрики RFC Java-застосунків ґрунтується на підході, описаному в [12], що використовує багатовимірні нормалізуючі перетворення та побудову прогнозних інтервалів. Методика включає кілька послідовних етапів.

- *Перший етап*: вихідні дані перевіряються на наявність багатовимірних аномалій. Якщо такі спостереження виявлено, їх вилучають із вибірки. Перевірка проводиться за критерієм Махаланобіса з порогом значущості 0,005 для нормалізованих даних [12]
- *Другий етап*: формується нелінійна регресійна модель шляхом застосування трансформації Box–Cox [11] або Johnson [12]. Аналіз виконується у нормалізованому просторі, після чого модель повертається до початкової шкали через обернене перетворення.
- *Третій етап*: визначаються межі прогнозування при рівні значущості 0,05, що дозволяє отримати інтервали, у яких із певною ймовірністю може перебувати прогнозоване значення RFC [11, 12].
- *Четвертий етап*: перевіряється, чи не виходять фактичні спостереження за межі прогнозних інтервалів. Якщо виходять – вони вважаються викидами, вилучаються з даних, і алгоритм повторюється для оновленої вибірки, доки вона не стане статистично однорідною. Якщо викидів немає, процес завершується, і модель визнається побудованою [2, 3].

Таким чином, застосування описаної методики забезпечує створення надійної нелінійної регресійної моделі взаємозв’язку між метриками RFC,

CBO та WMC. Подібні моделі особливо корисні для оцінки структурної складності та якості архітектури Java-систем у сфері e-commerce, де точність вимірювання цих показників значною мірою визначає масштабованість, стабільність та продуктивність програмних платформ.

1.3 Обґрунтування необхідності проведення досліджень за обраною темою

У контексті дослідження якості програмних систем, зокрема веб-орієнтованих застосунків у сфері електронної комерції, реалізованих мовою Java, визначальною умовою побудови коректних математичних моделей виступає достовірність та надійність вихідних даних, що залучаються для процесів навчання, налаштування й експериментальної перевірки. Серед численних показників об’єктно-орієнтованого програмування особливе значення мають метрики RFC, CBO та WMC. Вони формують основу багатовимірного статистичного простору, який відображає рівень структурної складності, ступінь зв’язаності та функціональне наповнення класів і компонентів програмної системи. Зазначені метрики активно застосовуються як індикатори ймовірності виникнення дефектів, накопичення технічного боргу та ускладнень у процесі супроводження програмних систем середнього й великого масштабу [13].

Під час розроблення математичних моделей, призначених для автоматизації процесів оцінювання якості програмних систем у сфері електронної комерції, важливо враховувати, що вибірка даних, сформована на основі метрик RFC, CBO та WMC, утворює багатовимірний статистичний простір. У цьому просторі наявність викидів здатна істотно впливати на точність оцінювання параметрів моделі та знижувати її прогностичну ефективність [14, 15]. Джерела появи таких відхилень можуть бути різноманітними: похибки інструментального вимірювання метрик,

архітектурна неоднорідність між окремими модулями системи, а також відмінності у практиках програмної розробки різних команд. Ігнорування викидів створює ризик отримання некоректних висновків як у випадку лінійних, так і нелінійних регресійних моделей, оскільки їхні параметричні оцінки є чутливими до крайніх значень.

У практиці дослідження програмних систем електронної комерції метод лінійної регресії часто використовується як базовий інструмент для аналізу взаємозв’язків між показниками якості та залежними характеристиками, такими як кількість виявлених дефектів чи витрати на супровід. Його популярність пояснюється зрозумілою інтерпретацією параметрів та відносною простотою реалізації. Втім, припущення про лінійність кореляцій не завжди відповідає реальним умовам функціонування складних об’єктно-орієнтованих систем. У таких випадках взаємодія між окремими метриками може мати нелінійний характер, наприклад проявлятися у вигляді квадратичних чи експоненційних залежностей, що відображають комбінований вплив зв’язаності та функціональної складності. Для адекватного моделювання подібних закономірностей доцільним є застосування нелінійних регресійних методів або моделей зі змінною кількістю ступенів свободи [16]. Водночас такі підходи потребують ретельної підготовки даних, зокрема ідентифікації та корекції викидів, які здатні суттєво спотворювати параметричні оцінки та знижувати точність прогнозування.

Одним із методів, що має ґрунтовне статистичне підґрунтя для ідентифікації нетипових спостережень у багатовимірних вибірках, є відстань Махаланобіса. На відміну від евклідової метрики, яка розглядає дані без урахування їхньої внутрішньої кореляційної структури, підхід Махаланобіса здійснює нормалізацію простору з використанням коваріаційної матриці. Це забезпечує можливість виявлення спостережень, що істотно відхиляються від загальної закономірності розподілу даних [17]. Така властивість є особливо значущою для метрик RFC, CBO та WMC, які нерідко демонструють

взаємозалежність: наприклад, зростання рівня зв’язаності часто супроводжується підвищенням функціональної складності у класах із великою кількістю методів. У результаті формується складна коваріаційна структура, що потребує адекватних інструментів для її аналізу.

Таким чином, інтеграція метрик RFC, CBO та WMC із методами багатовимірного статистичного аналізу, застосуванням лінійних і нелінійних регресійних моделей та процедурами ідентифікації викидів постає як науково обґрунтований підхід, що відповідає сучасним критеріям оцінювання якості програмних систем. Побудова відповідної математичної моделі разом із програмним інструментарієм для її реалізації формує основу для створення об’єктивних, відтворюваних і автоматизованих механізмів підтримки прийняття рішень щодо якості Java-орієнтованих систем електронної комерції. Це, у свою чергу, визначає не лише наукову новизну дослідження, але й його прикладну цінність у реальних умовах проєктування та промислової експлуатації.

1.4 Висновки за розділом 1

У цьому розділі подано аналіз існуючих математичних моделей, що застосовуються для підтримки оцінювання характеристик якості Java-застосунків у сфері електронної комерції, а також обґрунтовано актуальність подальших досліджень, спрямованих на автоматизацію процесів прийняття рішень щодо якості таких програмних систем.

Показано, що на практиці для формування рішень щодо якості e-commerce систем на платформі Java переважно використовуються підходи, засновані на лінійних регресійних моделях із залученням обмеженого набору структурних метрик, зокрема CBO та WMC. Водночас подібні моделі не завжди забезпечують достатній рівень достовірності управлінських рішень, особливо для систем електронної комерції, які характеризуються складною

багаторівневою архітектурою та інтенсивною взаємодією між компонентами. Це зумовлює необхідність переходу до більш гнучких інструментів підтримки рішень, зокрема із використанням нелінійних регресійних залежностей.

Обґрунтовано доцільність застосування нелінійних регресійних моделей із нормалізуючими перетвореннями та елементами інтервального прогнозування як основи для автоматизованого прийняття рішень щодо якості Java-орієнтованих e-commerce програмних систем. Такий підхід дозволяє підвищити точність та надійність оцінювання комплексних характеристик якості, що створює передумови для ефективної автоматизації процесів аналізу, контролю та управління якістю відкритих програмних систем електронної комерції.

2 МАТЕМАТИЧНА МОДЕЛЬ ДЛЯ АВТОМАТИЗАЦІЇ ПРИЙНЯТТЯ РІШЕНЬ ЩОДО ЯКОСТІ ПРОГРАМНИХ СИСТЕМ E-COMMERCE НА JAVA

2.1 Аналіз наявних математичних моделей оцінювання якості програмних систем та вибір факторів для автоматизації прийняття рішень щодо якості e-commerce програмних систем на Java

Ефективна автоматизація процесів оцінювання якості програмних систем електронної комерції, реалізованих мовою Java, потребує створення цілісної методологічної бази.

Така база має поєднувати три взаємопов’язані рівні: концептуальний – що визначає сутність і критерії поняття «якість»; емпіричний – який спирається на вимірювані індикатори архітектурних та кодових характеристик; та аналітичний – що забезпечує математичне моделювання залежностей між цими індикаторами та цільовими показниками якості. У сучасній дослідницькій та інженерній практиці подібні системи формуються або на основі нормативних ієрархічних моделей, які задають структуру критеріїв, або шляхом побудови статистичних та машинно-навчальних предиктивних моделей, що ґрунтуються на емпіричних даних. Для завдань оцінювання якості Java-орієнтованих e-commerce систем доцільним є інтеграція обох підходів: нормативна модель визначає рамку та ієрархію критеріїв, тоді як регресійні та ймовірнісні методи забезпечують автоматизоване прогнозування ризиків і формування об’єктивних рішень на основі метрик програмного коду.

У ролі концептуальної «рамки» для оцінювання якості програмних продуктів часто використовується стандарт ISO/IEC 25010, який пропонує модель якості, структуровану через систему характеристик та підхарактеристик [18, 19]. Його практична значущість полягає у створенні

єдиної термінологічної основи та можливості формування ієрархії цілей оцінювання. Для систем електронної комерції, що характеризуються високим навантаженням, динамічними змінами бізнес-логіки та інтеграціями з платіжними й логістичними сервісами, ключовими стають такі властивості, як надійність, продуктивність, супроводжуваність і сумісність. Водночас стандарт ISO/IEC 25010 не визначає формальних математичних процедур трансформації метрик у кількісні показники якості, тому його застосування в автоматизованих системах підтримки рішень потребує адаптації до вимірюваних індикаторів та інтеграції зі статистичними моделями.

У межах ієрархічних концепцій оцінювання якості програмного забезпечення значну роль відіграють моделі, що встановлюють відповідність між низькорівневими показниками та високорівневими характеристиками. Одним із найбільш відомих прикладів є [19], який демонструє системний підхід до формалізації процесу оцінювання. У його основі лежить визначення ключових властивостей об’єктно-орієнтованого дизайну – таких як інкапсуляція, спадкування, зв’язність і поліморфізм – та розроблення механізму агрегування метрик у комплексні індикатори якості архітектури. Перевагою подібних моделей є їхня прозорість та можливість стандартизувати процедури аналізу, що робить їх корисними у дослідницькій та інженерній практиці. Водночас універсальність таких підходів є обмеженою: без адаптації до специфіки предметної області (наприклад, систем електронної комерції), технологічного середовища та архітектурних стилів результати можуть втратити релевантність, оскільки розподіл метрик значною мірою залежить від контексту реалізації.

Найбільш розповсюджений тип предиктивних моделей у сфері оцінювання якості програмного забезпечення базується на використанні метрик об’єктно-орієнтованого проєктування та програмного коду, зокрема на класичному CK-наборі, запропонованому Чідамбером і Кемерером. У фундаментальній праці [13] було сформульовано та емпірично підтверджено низку метрик, серед яких особливе значення для аналізу структурної

складності та взаємозалежностей мають WMC, CBO та RFC [13]. Для Java-орієнтованих систем ці показники є природними індикаторами, оскільки вони безпосередньо відображають властивості, що впливають на супроводжуваність та ймовірність виникнення дефектів. Зокрема, збільшення кількості або «ваги» методів ускладнює процес тестування та розуміння коду; високий рівень зв’язаності між класами підвищує ризик каскадних змін; а розширення множини можливих «відповідей» класу ускладнює перевірку його поведінки та спричиняє більшу варіативність виконання [13].

У сучасних дослідженнях, що базуються на метриках CK, значне поширення отримали статистичні моделі різних типів: лінійна та нелінійна регресія застосовуються для прогнозування кількісних показників якості, тоді як логістична регресія використовується для класифікації класів за ознакою схильності до дефектів. Емпіричні результати підтверджують наявність кореляцій між WMC, RFC та CBO і дефектністю програмних компонентів, проте водночас акцентують на впливі конфаундерів, насамперед ефекту розміру. При контролі цього фактора значущість окремих метрик може зникати, що вимагає ретельної постановки моделі та обережної інтерпретації результатів [20]. Для Java-орієнтованих систем електронної комерції ця проблема є особливо актуальною, адже їхні модулі можуть істотно відрізнятися за масштабом, і без нормалізації чи контролю розміру модель ризикує відображати лише ефект великих компонентів.

Паралельно розвиваються ймовірнісні та data mining-підходи, серед яких варто відзначити байєсівські мережі, що застосовуються для прогнозування дефектності на основі CK-метрик [21]. Їхньою перевагою є здатність враховувати нелінійні залежності та взаємодію факторів. Водночас у практиці управління проєктами часто пріоритетною залишається інтерпретованість моделей – можливість чітко пояснити причини віднесення компонента до ризикових. Саме тому регресійні моделі та прозорі правила агрегування метрик залишаються більш придатними для інтеграції у системи автоматизованої підтримки рішень.

З урахуванням існуючих підходів, побудова моделі автоматизованої підтримки рішень щодо якості програмних систем потребує ретельного відбору факторів, які одночасно:

- концептуальний зв’язок із ключовими атрибутами якості;
- можуть бути отримані автоматизовано шляхом статичного аналізу коду;
- залишаються стабільними та порівнюваними між різними компонентами системи;
- підтверджені емпіричними дослідженнями. Найбільш узгоджено цим критеріям відповідає тріада WMC–CBO–RFC, яка формує основу факторного простору.

Метрика WMC виступає індикатором складності та функціонального навантаження класу; у випадку e-commerce систем вона часто відображає концентровану бізнес-логіку (наприклад, правила знижок, умови доставки чи комбінаторні сценарії), що безпосередньо впливає на трудомісткість тестування та модифікації. CBO характеризує інтенсивність міжкласових залежностей, які у системах електронної комерції природно зростають через численні інтеграції (платіжні шлюзи, складські системи, CRM, аналітичні модулі). Надмірне значення CBO ускладнює локалізацію змін та підвищує ризик регресій. RFC відображає різноманітність можливих викликів і сценаріїв виконання; для високонавантажених Java-сервісів у сфері e-commerce це прямо корелює з вимогами до надійності та передбачуваності поведінки, адже зі зростанням множини відповідей зростає складність забезпечення повноти тестового покриття.

Таким чином, сучасні підходи до оцінювання якості програмного забезпечення формують логічно узгоджений ланцюг методологій. Метрико-орієнтовані моделі, такі як CK та QMOOD, конкретизують «чим вимірювати» структурні характеристики об’єктно-орієнтованих систем. Регресійні та ймовірнісні підходи, у свою чергу, задають «як саме» обчислювати прогнозні індекси якості, враховуючи емпіричні дані та статистичні ефекти.

Для Java-орієнтованих систем електронної комерції найбільш доцільним вихідним набором факторів виступає тріада WMC, CBO та RFC, оскільки ці метрики є інтерпретованими, автоматично вимірюваними та підтвердженими як теоретично, так і емпірично у зв’язку з ключовими атрибутами якості. Водночас побудова коректної моделі потребує врахування ефекту розміру та забезпечення статистичної узгодженості даних. Це створює основу для наступних етапів дослідження: відбору репрезентативних даних, виявлення та обробки викидів, розроблення лінійних і нелінійних регресійних моделей, а також формування формалізованих правил автоматизованої підтримки управлінських рішень.

2.2 Попередня обробка та аналіз емпіричних даних для побудови математичної моделі оцінювання якості програмних систем e-commerce, реалізованих мовою Java

Для реалізації цього завдання було сформовано вибірку з 36 відкритих Java-застосунків, розміщених на платформі GitHub [22]. Метрики RFC, CBO та WMC отримано за допомогою інструмента CK [23], який широко використовується для аналізу програмних характеристик.

Для побудови нелінійної регресійної моделі, призначеної для оцінювання метрики RFC програмних систем у сфері електронної комерції з відкритим кодом на Java, використовується вибірка тривимірних негаусових даних, наведених у табл. 2.1, де RFC позначено як (Y), а CBO та WMC – як (X_1) та (X_2).

З метою виявлення можливих викидів у сформованій вибірці було застосовано метод, що ґрунтується на використанні нормалізуючих перетворень і квадрата відстані Махаланобіса. Значення квадрата d_i^2 для кожної багатовимірної точки нормалізованих даних i, $i = 1,2,\dots,N$, визначались відповідно до формули:

$$d_i^2 = (Z_i - \bar{Z})^T S_N^{-1} (Z_i - \bar{Z}), \quad (2.1)$$

де $\bar{Z}$– вектор середніх значень вибірки для змінних X_1, X_2, Y;

Z_i– вектор спостережень (значення змінних X_1, X_2, Y для кожного застосунку);

S_N– коваріаційна матриця:

$$S_N = \frac{1}{N} \sum_{i=1}^{N} (Z_i - \bar{Z})(Z_i - \bar{Z})^T. \quad (2.2)$$

Таблиця 2.1 - Вибірка для побудови моделі

№	URL	Y	X_1	X_2
1	2	3	4	5
1	https://github.com/adityakaushal/MyKart	5,7022	1,2921	3,4607
2	https://github.com/ajaymahadeven/E-Commerce-Platform	14,6923	2,3846	26,2308
3	https://github.com/Ajmal0197/oMart	6,814	4,4884	3,093
4	https://github.com/Akshay-101096/E-commerce	6,5	1,8	4,3
5	https://github.com/Akxsh20/OneClickAway-E-Commerce	14,381	6,1429	11,4762
6	https://github.com/anirudhmsu/SneakSecure-microservices-framework	3,1176	5,3235	0,9118
7	https://github.com/bhagatanirudh/E-Commerce-Website	11,5588	3,9118	10,7941
8	https://github.com/CyrielleGl/KiraDesign	10,3284	7,2537	7,0597
9	https://github.com/EricJoseph98/Maverick-Essentials-Android-App	6,5181	3,8072	2,5422
10	https://github.com/imxushuai/leyou	5,8296	5,2815	3,5037
11	https://github.com/IncridableAcuman/fasion-commerce	5,0682	5,6364	2,4773
12	https://github.com/jigishasurani/e-commerce-application-Brusting-basket	6,4318	4,5455	3
13	https://github.com/madhurajayashanka/ShopBeastBackend	5,2308	6,6154	3
14	https://github.com/Mandar800/E-Farm	7,3871	4,1505	3,0323
15	https://github.com/marwa-eltayeb/Souq_ShopOnline	8,8571	6,44	4,4
16	https://github.com/MathDEV-0/Quickbite-FoodDelivery	9,0227	7,25	7,2955
17	https://github.com/MrDiipo/ECommerce-Event-Driven-Microservices-System	2,5	3,8333	1,75
18	https://github.com/mustafaynk/shoearstore	8,3246	4,2761	6,4552
19	https://github.com/nihadamirov/e-commerce-platform	5,6098	7,2561	2,5488
20	https://github.com/omarr45/FlyBuy	7,931	5,7586	4,3966
21	https://github.com/plutonicdev/GroceryStore-with-server	10,2717	6,3121	5,7341
22	https://github.com/pranjalsharma26/E-Commerce-Android-Application	7,052	4,8266	3,1214
23	https://github.com/preetikumari18/E-Commerce-Website	12,48	4,48	10,6
24	https://github.com/RaefGaied/quickbuy-ecommerce	13,8148	5,7407	13,7778
25	https://github.com/rciphertext/zkart	4,1111	2,3333	5,3333

Продовження таблиці 2.1

1	2	3	4	5
26	https://github.com/RohitBhandare/ecommerce-webapp-hibernate-servlet	7,5385	5,4615	6,3846
27	https://github.com/sametakbal/kips-shop	4,8857	6,6571	2,1429
28	https://github.com/saurabh-maurya1/BackendCode_Project	6,8235	8,6176	2,9412
29	https://github.com/shadow0403bsr/Price-Comparison-Engine	16,6014	7,1104	29,1408
30	https://github.com/simpledj/McDonald-Delivery-App	3,7143	3,2857	1
31	https://github.com/simpledj/Penshoppe-Men-App-with-Firebase-System	6,4051	4,4051	2,8608
32	https://github.com/trong0dn/Tea-Shoppe	9,45	2,1	9,85
33	https://github.com/Vzenun/E-Commerce_Application-JAVA-	22,7143	3	29,8571
34	https://github.com/WDpad753/E-Commerece-application	8,6591	1,3409	4,7045
35	https://github.com/WesleyKaihara/3dlab_backend	6,6786	6,8214	5,9643
36	https://github.com/YunussEmree/buyer	5,3429	6,4429	3,5286

Результати аналізу тривимірних даних, нормалізованих за допомогою десяткового логарифму, засвідчили наявність семи статистично значущих викидів у багатовимірному просторі факторів при рівні значущості α = 0,005. Ідентифікація аномальних спостережень здійснювалася на основі аналізу квадрата відстані Махаланобіса d_i^2 з порівнянням максимальних значень з критичним значенням χ^2-розподілу для трьох ступенів вільності, яке становить $\chi^2_{0,995} = 12{,}838$.

На першому етапі аналізу для початкової вибірки обсягом $n = 36$ було отримано максимальне значення квадрата відстані Махаланобіса $D_i^2\{\max\}$ = 11,224, що не перевищувало критичне значення. Проте після побудови початкової регресійної моделі найбільший залишковий викид було зафіксовано у 25-му рядку, що стало підставою для його вилучення. Наступна ітерація аналізу при $n = 35$: виявили перевищення критичного рівня χ^2з максимальними значеннями $D_i^2\{\max\}$ = 13,968 та ідентифікацією викида у 17-му рядку.

На наступних етапах очищення вибірки при $n = 34$, $n = 33$, $n = 32$, $n = 31$ та $n = 30$ максимальні значення квадрата відстані Махаланобіса становили 8,245, 8,173, 8,256, 8,310 та 8,264, що формально не

перевищували критичне значення χ^2. Однак аналіз найбільших залишків лінійної регресійної моделі дозволив ідентифікувати додаткові аномальні спостереження у 35-му, 36-му, 13-му, 26-му та 33-му рядках. Таким чином, загалом з початкової вибірки було вилучено сім тривимірних викидів, що відповідають рядкам 25, 17, 35, 36, 13, 26 та 33.

Після вилучення зазначених аномальних спостережень остаточна скоригована вибірка склала $n = 29$ записів, що становить 80,6% від початкового обсягу даних. Повторна перевірка статистичних характеристик скоригованої вибірки за допомогою тесту Мардіа показала суттєве зменшення показників асиметрії та ексцесу багатовимірного розподілу, що свідчить про наближення логарифмічно нормалізованих даних до багатовимірного нормального розподілу.

Використання очищеної вибірки при побудові нелінійної регресійної моделі оцінювання метрики RFC програмних систем електронної комерції з відкритим кодом, розроблених мовою Java, забезпечило істотне підвищення якості моделювання.

2.3 Побудова математичної моделі для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java

Для побудови нелінійної регресійної моделі для оцінювання метрики RFC застосунків з відкритим кодом на Java, ми використовуємо метод, що заснований на нормалізуючих перетвореннях та інтервалах прогнозування, а також модифікований метод, який додатково враховує можливу наявність викидів у нелінійній регресійній моделі. У загальному випадку процес побудови нелінійної регресійної моделі за цим методом є ітераційним.

Як було зазначено раніше, у вихідних даних з табл. 2.1, що містять 36 тривимірних спостережень, у процесі ітеративного аналізу було виявлено сім тривимірних викидів. Зокрема, за результатами аналізу квадрата відстані

Махаланобіса та найбільших залишків регресійної моделі, до аномальних спостережень було віднесено рядки 25, 17, 35, 36, 13, 26 та 33 табл. 2.1. Виявлення зазначених викидів здійснювалося послідовно на різних ітераціях побудови моделі.

Далі, відповідно до обраного методу, на черговій ітерації тривимірні негаусівські дані з табл. 2.1 (без рядків 25, 17, 35, 36, 13, 26 та 33) знову нормалізуються за допомогою нормалізуючого перетворення у формі десяткового логарифму. При цьому тривимірних викидів у нормалізованих даних не було виявлено. Викиди відсутні, оскільки всі значення квадрату відстані Махаланобіса у цьому випадку є меншими за значення квантилю розподілу χ^2, який дорівнює 12,84 для трьох ступенів свободи та рівня значущості 0,005.

Після вилучення зазначених аномальних спостережень та нормалізації даних, для скоригованої вибірки, що складається з 29 рядків, будуємо лінійну регресійну модель для визначення нормалізованого значення метрики RFC у вигляді

$$Z_Y = \hat{Z}_Y + \varepsilon = \hat{b}_0 + \hat{b}_1 Z_1 + \hat{b}_2 Z_2 + \varepsilon, \quad (2.3)$$

де: Z_Y – нормалізоване значення метрики RFC;

Z_1та Z_2 – нормалізовані значення метрик CBO та WMC відповідно;

$\hat{b}_0, \hat{b}_1, \hat{b}_2$ – оцінки параметрів лінійної регресійної моделі;

ε– випадкова похибка.

Оцінка параметрів моделі b_0, b_1, b_2 здійснюється методом найменших квадратів:

$$\hat{b} = (X^T X)^{-1} X^T y, \quad (2.4)$$

де $\hat{b}$ – оцінка параметрів методом найменших квадратів;

X– матриця регресорів;

y– вектор значень змінної відгуку.

Після виконання обчислень для скоригованої вибірки, що складається з 29 спостережень, були отримані такі значення коефіцієнтів: $\hat{b}_0 = 0{,}5274, \hat{b}_1 = 0{,}0739, \hat{b}_2 = 0{,}4719,$ де $\varepsilon -$ гаусівська випадкова величина з нульовим математичним сподіванням. Оцінка середньоквадратичного відхилення залишків випадкової величини ε дорівнює $0{,}048142.$

За допомогою зворотного перетворення отримаємо модель нелінійної регресії. Формуємо нелінійну регресійну модель з урахуванням зворотного логарифмічного перетворення:

$$Y = 10^{\varepsilon+\hat{b}_0} X_1^{\hat{b}_1} X_2^{\hat{b}_2}. \tag{2.5}$$

Далі, визначаємо границі інтервалу прогнозування нелінійної регресії на нормалізованих даних. Границі інтервалу прогнозування (на логарифмічній шкалі) для рівня значущості $\alpha = 0{,}05$ обчислюємо формулою:

$$\psi_Y^{-1}\left(\hat{Z}_Y \pm t_{\alpha/2,\nu}\, S_{Z_Y} \sqrt{1 + \frac{1}{N} + (z_X^+)^T ((Z_X^+)^T Z_X^+)^{-1} (z_X^+)}\right) \tag{2.6}$$

де: ψ_Y^{-1}– зворотне до логарифмічного перетворення;

$\hat{Z}_Y$– прогноз лінійної регресії для нормалізованих значень факторів;

$t_{\alpha/2,\nu}$– квантиль t-розподілу Стьюдента;

S_{Z_Y}– оцінка стандартного відхилення залишків;

N– обсяг вибірки на поточній ітерації;

z_X^+– вектор центрованих нормалізованих незалежних змінних;

$(Z_X^+)^T Z_X^+$– матриця центрованих нормалізованих незалежних змінних.

Матриця має вигляд:

$$(Z_X^+)^T Z_X^+ = \begin{pmatrix} S_{Z_1 Z_1} & S_{Z_1 Z_2} \\ S_{Z_1 Z_2} & S_{Z_2 Z_2} \end{pmatrix}, \quad (2.7)$$

В результаті була побудована двохфакторна нелінійна регресійна модель, яка має наступний вигляд:

$$\mathrm{Y} = 10^{\varepsilon + 0{,}5274} \mathrm{X}_1^{0{,}0739} \mathrm{X}_2^{0{,}4719}.$$

Для цієї моделі матриця центрованих нормалізованих незалежних змінних:

$$\begin{pmatrix} 0{,}7544 & 0{,}0117 \\ 0{,}0117 & 0{,}2807 \end{pmatrix}.$$

Параметри якості моделі були перевірені за такими показниками: R^2, *MMRE*, *Pred*(0,25).

Зокрема, для фінальної моделі отримано коефіцієнт детермінації $R^2 = 0{,}905$ та скоригований коефіцієнт $R^2_{adj} = 0{,}898$, середню відносну похибку прогнозування $MMRE = 0{,}0959$ та показник $Pred(0{,}25) = 1{,}0$, що відповідає 29 коректним прогнозам із 29 можливих. Це підтверджує доцільність застосування процедури виявлення та вилучення багатовимірних викидів перед автоматизацією прийняття рішень щодо якості e-commerce програмних систем на платформі Java.

У таблиці 2.2 наведено значення Y(RFC), передбачені значення $\hat{Y}$, а також розраховані межі довірчого інтервалу та інтервалу прогнозування для кожного застосунку.

Таблиця 2.2 - Передбачені значення та інтервали

index	Y	$\hat{Y}$	Confidence low	Confidence upper	Prediction low	Prediction upper
1	2	3	4	5	6	7
1	5,7022	6,1662	5,4702	6,9508	4,7305	8,0378

Продовження таблиці 2.2

1	2	3	4	5	6	7
2	14,6923	16,7803	14,9407	18,8465	12,8942	21,8378
3	6,8140	6,4120	6,1030	6,7366	5,0359	8,1639
4	6,5000	7,0011	6,3879	7,6733	5,4328	9,0221
5	14,3810	12,1836	11,3335	13,0974	9,5145	15,6014
6	3,1176	3,6487	3,3005	4,0336	2,8222	4,7172
7	11,5588	11,4478	10,7392	12,2032	8,9608	14,6251
8	10,3284	9,8070	9,1763	10,4811	7,6712	12,5375
9	6,5181	5,7745	5,4551	6,1126	4,5278	7,3644
10	5,8296	6,8828	6,5528	7,2296	5,4061	8,7631
11	5,0682	5,8724	5,5323	6,2334	4,6015	7,4942
12	6,4318	6,3262	6,0164	6,6518	4,9677	8,0560
14	7,3871	6,3156	6,0059	6,6413	4,9594	8,0428
15	8,8571	7,7771	7,3589	8,2191	6,1004	9,9147
16	9,0227	9,9599	9,3135	10,6511	7,7894	12,7351
18	8,3246	9,0411	8,6233	9,4790	7,1038	11,5066
19	5,6098	6,0640	5,6540	6,5036	4,7387	7,7989
20	7,9310	7,7103	7,3351	8,1047	6,0550	9,8180
21	10,2717	8,7993	8,3250	9,3007	6,9020	11,2182
22	7,0520	6,4744	6,1605	6,8042	5,0847	8,2439
23	12,4800	11,4646	10,7703	12,2037	8,9772	14,6412
24	13,8148	13,2148	12,2315	14,2771	10,3042	16,9474
27	4,8857	5,5519	5,1732	5,9583	4,3377	7,1059
28	6,8235	6,5709	6,0800	7,1015	5,1232	8,4278
29	16,6014	19,1183	16,9980	21,5031	14,6814	24,8962
30	3,7143	3,6776	3,3308	4,0605	2,8459	4,7523
31	6,4051	6,1716	5,8619	6,4976	4,8450	7,8613
32	9,4500	10,4710	9,5836	11,4405	8,1344	13,4786
34	8,6591	7,1472	6,3697	8,0197	5,4943	9,2975

Згідно з [1], якщо точка даних і для застосунку знаходиться в межах довірчого інтервалу, то програма має середню якість (Medium). Якщо точка даних і для застосунку знаходиться між межами довірчого інтервалу та інтервалу прогнозування, то програма має високу якість (High). Якщо точка даних і для застосунку вийшла за межі інтервалу прогнозування, то програма має низьку якість (Low).

Визначимо якість наступного застосунку: ShopBeastBackend (https://github.com/madhurajayashanka/ShopBeastBackend), RFC=5,2308, CBO=6,6154, WMC=3.

Розраховане значення Y згідно нелінійної регресійної моделі склало 6,5042, границі довірчого інтервалу становлять [6,1173; 6,9155], границі інтервалу прогнозування становлять [5,0945; 8,3039], тобто значення RFC, яке дорівнює 5,2308, потрапляє між нижніми межами довірчого інтервалу та інтервалу прогнозування і цей застосунок має високу якість (High).

2.4 Висновки за розділом 2

У цьому розділі проведено перевірку вихідних даних, призначених для побудови нелінійної регресійної моделі оцінювання метрики RFC для Java-застосунків з відкритим кодом, на наявність тривимірних викидів, а також виконано побудову самої нелінійної регресійної моделі для визначення значень метрики RFC таких застосунків.

Для перевірки даних застосовано комбінацію нормалізуючого перетворення та критерію квадрата відстані Махаланобіса, що дозволило обґрунтовано ідентифікувати аномальні спостереження у багатовимірному просторі метрик. Аналіз показав, що після нормалізації тривимірна вибірка містить сім чітко визначених багатовимірних викиди. Це підтверджує необхідність використання нормалізуючих перетворень як базового етапу підготовки даних у задачах автоматизованого аналізу якості e-commerce систем.

Запропонована удосконалена двофакторна нелінійна регресійна модель, побудована на основі метрик RFC, CBO та WMC із застосуванням логарифмічного перетворення та врахуванням викидів, забезпечує більш інформативну основу для прийняття управлінських рішень. Порівняно з моделями, що не враховують аномальні значення, вона характеризується підвищеними значеннями коефіцієнта детермінації та показників прогнозування, зменшенням середньої відносної похибки, а також звуженням довірчих і прогнозних інтервалів для більшості значень структурних метрик.

Отримані результати демонструють, що застосування нормалізуючих перетворень та врахування багатовимірних викидів є критичною умовою побудови адекватних нелінійних регресійних моделей, орієнтованих на автоматизацію оцінювання якості Java-орієнтованих e-commerce систем. Такий підхід підвищує достовірність аналітичних висновків і формує методологічне підґрунтя для ефективного управління якістю складних програмних продуктів у сфері електронної комерції.

3 РОЗРОБКА ПРОЄКТНИХ РІШЕНЬ ДЛЯ ПРОГРАМНОЇ СИСТЕМИ АВТОМАТИЗАЦІЇ ПРИЙНЯТТЯ РІШЕНЬ ЩОДО ЯКОСТІ ПРОГРАМНИХ СИСТЕМ E-COMMERCE, РЕАЛІЗОВАНИХ МОВОЮ JAVA

3.1 Загальносистемні проєктні рішення для програмної системи оцінювання якості e-commerce ПС на Java

Виходячи з результатів аналізу наявних моделей оцінювання метрики RFC для застосунків з відкритим кодом на Java, сформульовано постановку задачі дослідження. Метою є розроблення програмної системи для оцінювання метрики RFC Java-застосунків з відкритим кодом, який забезпечував би автоматизацію прийняття рішень щодо якості програмних систем, зокрема у сфері електронної комерції.

Вимоги до програми.

Вимоги до функціональних характеристик

Програмна система повинна реалізовувати визначений набір функцій, необхідних для коректного виконання процесу оцінювання метрики RFC та аналізу якості програмних систем.

Вимоги до складу виконуваних функцій:

Програмний засіб має забезпечувати виконання таких основних функцій:

1. Зчитування з текстового файлу емпіричних даних метрик RFC, CBO та WMC Java-застосунків з відкритим кодом;
2. Введення через інтерфейс програми значення довірчої ймовірності, а також значень метрик CBO та WMC для програмного застосунку, для якого необхідно виконати оцінювання метрики RFC;
3. Виконання нормалізації емпіричних даних із застосуванням нормалізуючого перетворення у формі десяткового логарифму;

4. Визначення значень коефіцієнтів лінійного регресійного рівняння для нормалізованих емпіричних даних;

5. Побудову нелінійної регресійної моделі для оцінювання метрики RFC;

6. Обчислення значень коефіцієнта детермінації R^2 та показників відносної похибки прогнозування, зокрема величини MMRE;

7. Формування довірчого інтервалу та інтервалу прогнозування для нелінійної регресійної моделі;

8. Виведення на екран результатів оцінювання, зокрема нелінійної регресійної залежності метрики RFC, відповідних довірчих та прогнозних інтервалів, значень коефіцієнта детермінації R^2 і показників відносної похибки;

9. Збереження у текстовому файлі результатів оцінювання метрики RFC Java-застосунків з відкритим кодом для подальшого аналізу та документування.

3.1.1 Вибір засобів та нотацій моделювання для програмної системи оцінювання якості

У процесі проєктування програмної системи для оцінювання якості e-commerce застосунків на Java важливим завданням є вибір нотацій моделювання, здатних забезпечити формалізований, однозначний і структурований опис архітектури та поведінки системи. Серед поширених підходів використовуються структурні нотації (DFD, IDEF0), об’єктно-орієнтовані засоби моделювання (UML) та архітектурні мови опису (ArchiMate).

DFD і IDEF0 ефективні для аналізу потоків даних і бізнес-процесів, проте їх функціональна спрямованість не дозволяє повноцінно відобразити об’єктну структуру та взаємодію компонентів складних Java-систем. ArchiMate забезпечує високорівневий опис архітектури та її зв’язку з

бізнес-контекстом, але є надто абстрактною для моделювання алгоритмів обчислення метрик і сценаріїв взаємодії користувача.

Найбільш придатною нотацією для проєктування системи оцінювання якості є UML, яка поєднує універсальність, формальність і наочність. UML дозволяє описати функціональні вимоги, взаємодію компонентів, поведінкові аспекти та сценарії роботи користувача. Крім того, UML є стандартом де-факто у проєктуванні Java-орієнтованих систем, що полегшує подальшу реалізацію та супровід програмного продукту.

Отже, UML доцільно обрати як основну нотацію моделювання, оскільки вона найповніше відповідає вимогам предметної області та забезпечує цілісне представлення функціональних і поведінкових характеристик системи.

3.1.2 Розробка діаграми варіантів використання програмної системи автоматизації прийняття рішень щодо якості

Діаграма варіантів використання програмної системи оцінювання якості e-commerce застосунків на Java відображає ключові функціональні можливості, доступні кінцевому користувачу. Центральним актором у моделі виступає користувач, який взаємодіє із системою з метою проведення оцінювання якості програмних продуктів. Виявлені варіанти використання наведені у таблиці 3.1.

Таблиця 3.1 - Виявлені варіанти використання

Номер варіанту використання	Основні актори	Найменування	Формулювання
A1	Користувач	Ввести дані метрик застосунків	Вводить емпіричні дані метрик RFC, CBO та WMC e-commerce застосунків з відкритим кодом на Java
A2	Користувач	Ввести метрики CBO та WMC застосунку	Вводить значення метрик CBO та WMC e-commerce застосунку, для якого необхідно здійснити оцінювання метрики RFC
A3	Користувач	Ввести довірчу ймовірність	Вводить значення довірчої ймовірності, необхідної для побудови довірчого інтервалу та інтервалу прогнозування нелінійної регресійної моделі оцінювання метрики RFC
A4	Користувач	Нормалізувати дані	Ініціює виконання нормалізації введених даних із застосуванням нормалізуючих перетворень
A5	Користувач	Побудувати нелінійну регресійну модель	Ініціює побудову нелінійної регресійної моделі для оцінювання значення метрики RFC e-commerce застосунків з відкритим кодом на Java
A6	Користувач	Отримати довірчий інтервал та інтервал прогнозування	Отримує для заданого рівня довірчої ймовірності довірчий інтервал та інтервал прогнозування нелінійної регресійної моделі для оцінювання метрики RFC

На рисунку 3.1 наведено діаграму варіантів використання програми для оцінювання метрики RFC застосунків з відкритим вихідним кодом, реалізованих мовою Java.

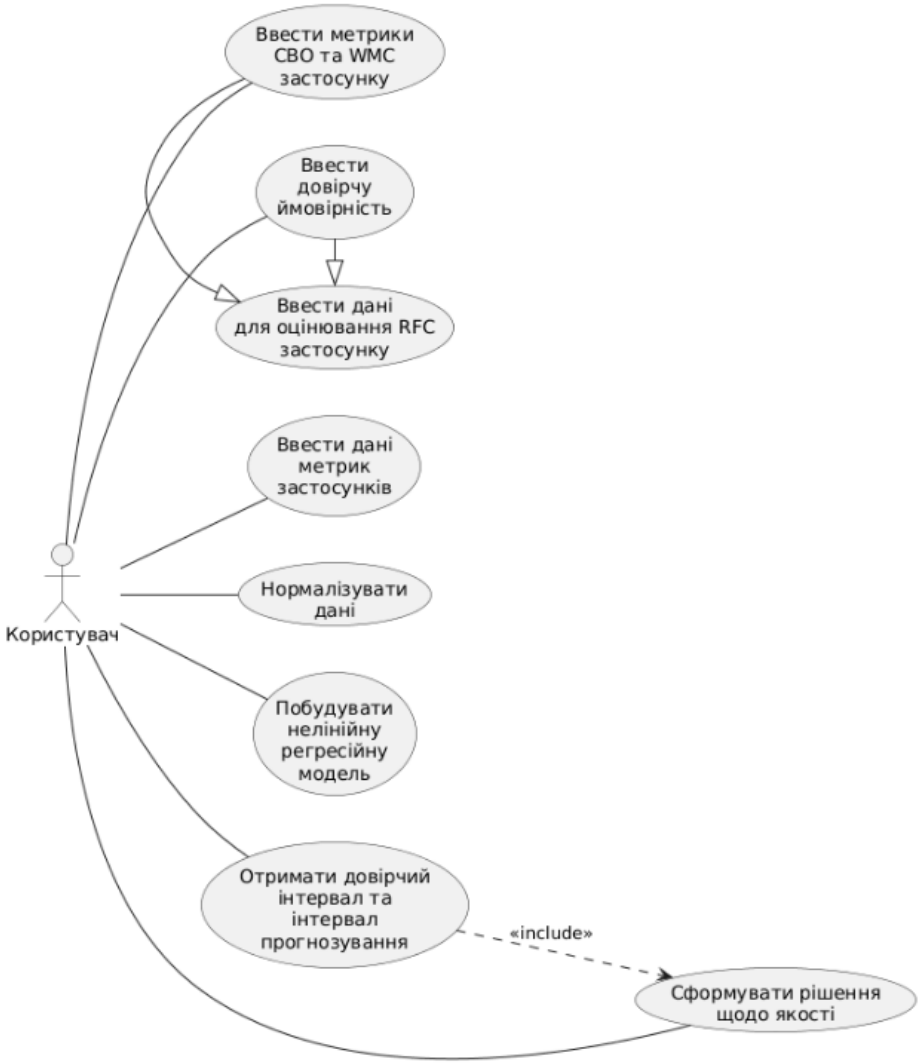

Рисунок 3.1 – Діаграма варіантів використання програмної системи для оцінювання метрики RFC застосунків e-commerce на Java

Отже, діаграма варіантів використання формує узагальнену картину функціональних можливостей програмної системи та відображає логіку взаємодії користувача з її основними сценаріями.

3.1.3 Специфікації варіантів використання програмної системи оцінювання якості e-commerce систем

Для кожного ключового варіанта використання формується окрема специфікація, у якій визначаються мета, передумови, очікуваний результат та обмеження виконання відповідного сценарію. Зокрема, для варіанта «Оцінювання якості e-commerce програмної системи» передбачається, що користувач заздалегідь підготував значення метрик RFC, CBO та WMC, отримані шляхом аналізу вихідного коду Java-застосунку.

A1 Введення емпіричних даних метрик

Дійова особа: Користувач.

Інші учасники: –

Короткий опис: Варіант використання дозволяє користувачеві ввести або завантажити з текстового файлу емпіричні дані метрик RFC, CBO та WMC e-commerce застосунків з відкритим кодом на Java для подальшого аналізу.

A2 Введення значень метрик CBO та WMC для оцінювання

Дійова особа: Користувач.

Інші учасники: –

Короткий опис: Варіант використання дозволяє користувачеві ввести значення метрик CBO та WMC конкретного e-commerce застосунку, для якого необхідно здійснити оцінювання метрики RFC.

A3 Введення довірчої ймовірності

Дійова особа: Користувач.

Інші учасники: –

Короткий опис: Варіант використання дозволяє користувачеві задати значення довірчої ймовірності, яке використовується для побудови довірчого

інтервалу та інтервалу прогнозування нелінійної регресійної моделі оцінювання метрики RFC.

А4 Нормалізація даних

Дійова особа: Користувач.

Інші учасники: –

Короткий опис: Варіант використання дозволяє користувачеві ініціювати нормалізацію введених емпіричних даних метрик RFC, CBO та WMC із застосуванням нормалізуючого перетворення у формі десяткового логарифму.

А5 Побудова нелінійної регресійної моделі

Дійова особа: Користувач.

Інші учасники: –

Короткий опис: Варіант використання дозволяє користувачеві побудувати нелінійну регресійну модель для оцінювання метрики RFC e-commerce застосунків з відкритим кодом на Java на основі нормалізованих даних.

А6 Отримання довірчого інтервалу та інтервалу прогнозування

Дійова особа: Користувач.

Інші учасники: –

Короткий опис: Варіант використання дозволяє користувачеві отримати довірчий інтервал та інтервал прогнозування нелінійної регресійної моделі для оцінювання значення метрики RFC з заданою довірчою ймовірністю.

Результатом виконання даного варіанта використання є формування числових показників якості програмної системи та отримання рекомендацій щодо її архітектурного стану, визначених на основі регресійної моделі й показників точності прогнозування. Специфікації варіантів використання

забезпечують формалізований опис вимог до системи та слугують методичною основою для її подальшої розробки.

3.1.4 Сценарії основних варіантів використання програмної системи оцінювання якості

Сценарії варіантів використання визначають послідовність дій користувача та відповідні реакції системи. Нижче представлено дві діаграми діяльності, рисунок 3.2 та рисунок 3.3.

Рисунок 3.2 – Діаграма діяльності для введення емпіричних даних метрик

Рисунок 3.3 – Діаграма діяльності для отримання довірчого інтервалу та інтервалу прогнозування

У типовому сценарії оцінювання якості Java-орієнтованого e-commerce застосунку користувач запускає процес, вводить або імпортує значення метрик, після чого система здійснює перевірку даних, виконує нормалізацію та формує нелінійну регресійну модель.

3.2 Проєктні рішення з інформаційного забезпечення програмної системи оцінювання якості

Ефективність роботи програмної системи оцінювання якості Java-орієнтованих e-commerce застосунків значною мірою визначається якістю організації інформаційного забезпечення. Воно формує склад, структуру та взаємозв’язки даних, що використовуються на всіх етапах автоматизованого прийняття рішень – від збору метрик до формування інтегральної оцінки. У процесі проєктування особливий акцент зроблено на створенні концептуальної та логічної моделей даних, які гарантують цілісність, узгодженість і масштабованість інформаційної бази.

Ключовою сутністю концептуальної моделі є e-commerce програмна система, для якої здійснюється оцінювання якості. З нею пов’язуються набори метрик, що містять значення RFC, CBO та WMC, отримані на певному етапі аналізу вихідного коду. Дані метрик можуть підлягати нормалізації та перевірці на наявність викидів, що обумовлює наявність відповідних концептуальних об’єктів, пов’язаних із процедурами обробки даних.

У логічній моделі даних програмної системи оцінювання якості e-commerce програмних систем на Java передбачено зберігання інформації про програмні системи, набори зібраних метрик та їх значення. Для забезпечення коректного аналізу даних виділено окремі структури для збереження нормалізованих значень метрик і результатів перевірки на викиди методом квадрата відстані Махаланобіса. Це дозволяє фіксувати не лише поточний стан даних, а й результати їх попередньої обробки.

3.2.1 Концептуальна модель даних програмної системи оцінювання якості e-commerce ПС

Концептуальна модель даних відображає узагальнене уявлення про предметну область без деталізації способів зберігання та реалізації даних. У межах програмної системи оцінювання якості e-commerce ПС на Java концептуальна модель даних описує основні сутності та їхні логічні взаємозв’язки, які виникають у процесі аналізу метрик і побудови регресійних моделей.

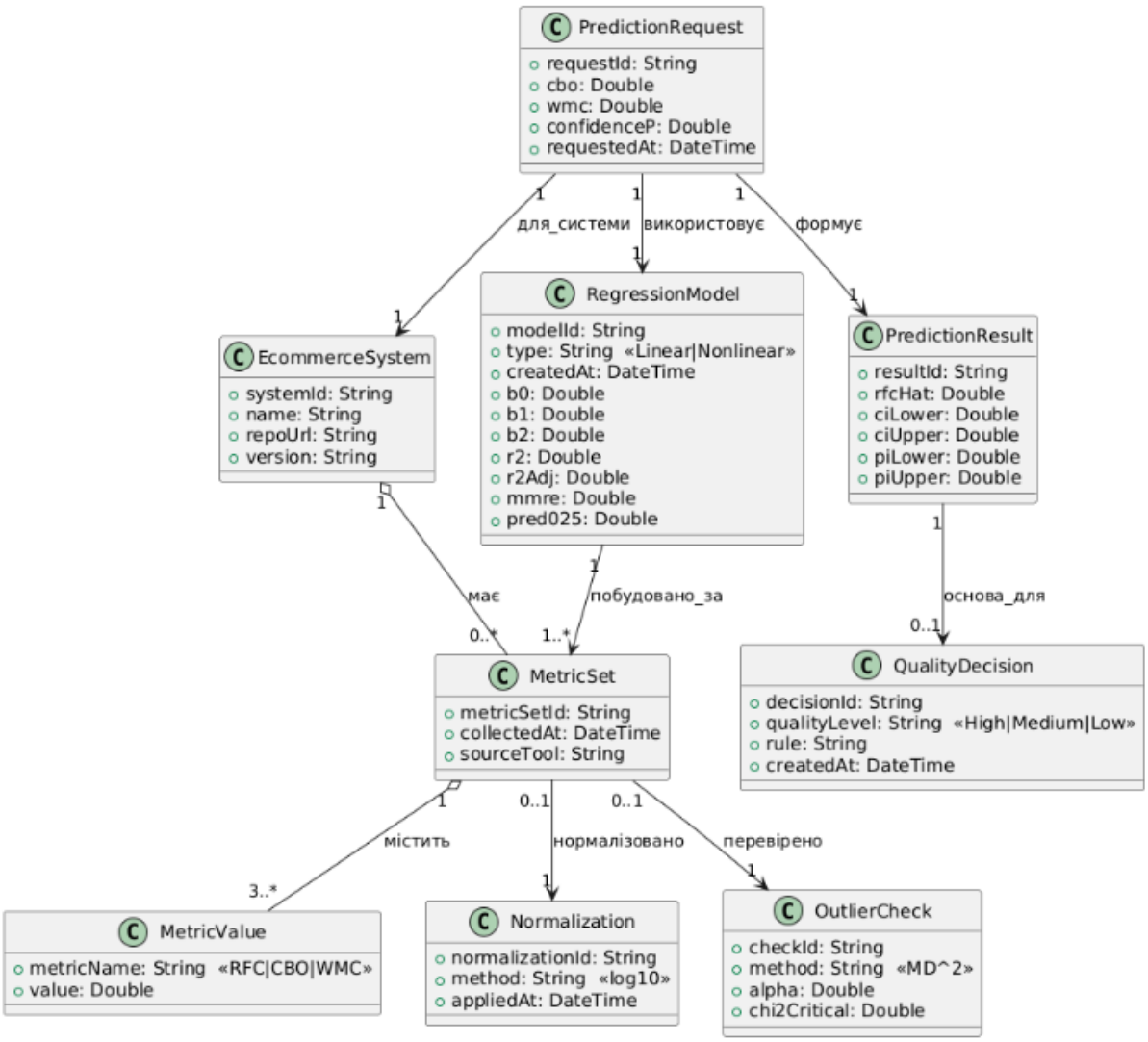

Рисунок 3.4 – Концептуальна модель даних програмної системи оцінювання якості e-commerce ПС

Таким чином, концептуальна модель даних визначає базові інформаційні об’єкти програмної системи оцінювання якості e-commerce ПС на Java та логічні зв’язки між ними, не прив’язуючись до конкретної реалізації сховища даних.

3.2.2 Логічна модель даних програмної системи оцінювання якості

Логічна модель даних є деталізованим поданням концептуальної моделі та призначена для формального опису структури даних на рівні логічної організації бази даних. Вона визначає склад сутностей, їхні атрибути, первинні та зовнішні ключі, а також кардинальності зв’язків між сутностями.

Логічна модель даних забезпечує структуроване та узгоджене зберігання інформації, необхідної для роботи програмної системи оцінювання якості Java-орієнтованих e-commerce застосунків, формує основу для фізичної реалізації бази даних і підтримує цілісність інформаційних процесів.

Логічна модель даних програмної системи оцінювання якості e-commerce ПС наведена на рисунку 3.5.

Загалом логічна модель даних забезпечує цілісне та структуроване зберігання всієї інформації, необхідної для функціонування програмної системи оцінювання якості e-commerce програмних систем на Java, створює основу для подальшої фізичної реалізації бази даних та гарантує узгодженість інформаційних процесів у системі.

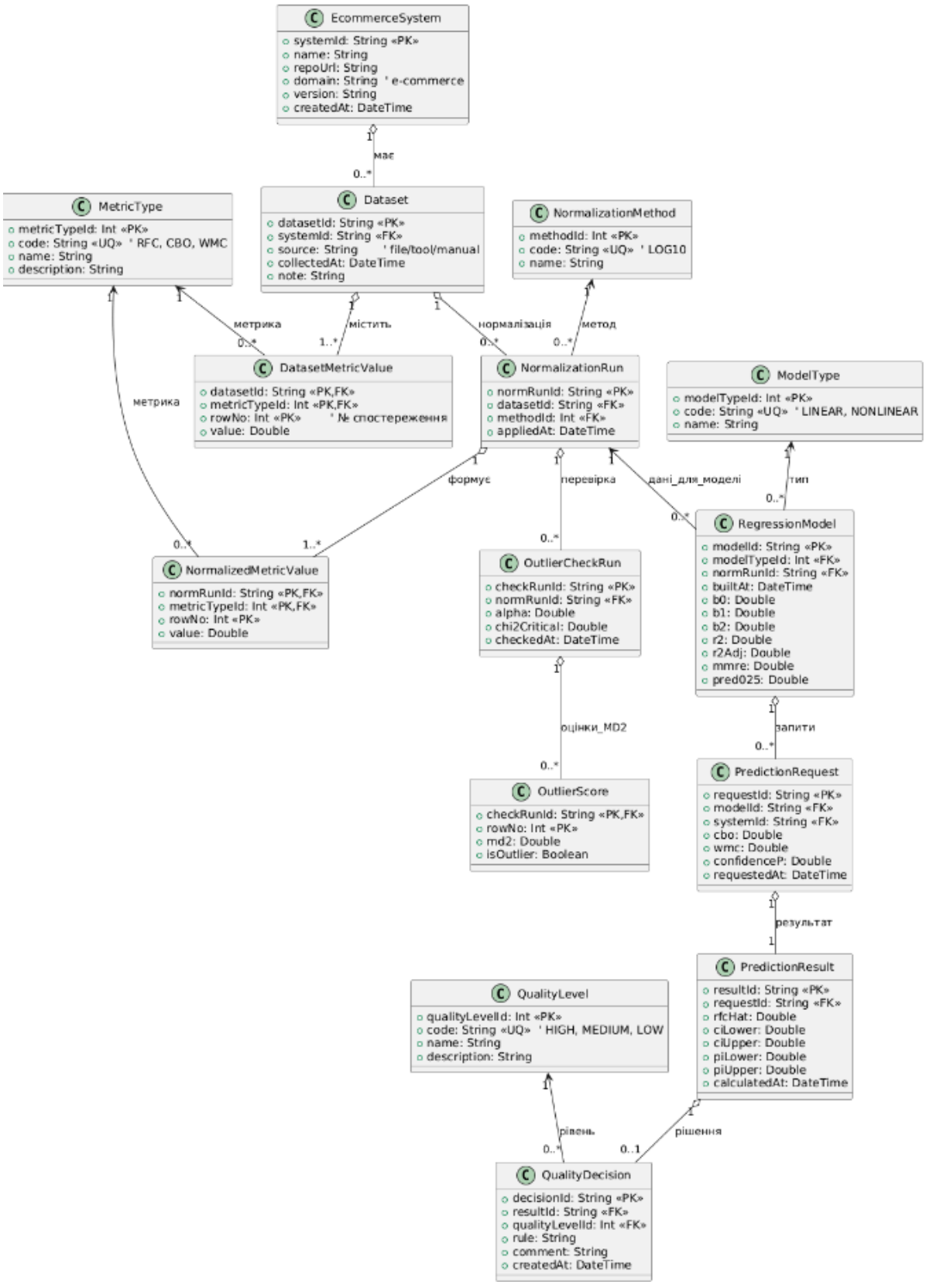

Рисунок 3.5 – Логічна модель даних програмної системи оцінювання якості e-commerce ПС

3.3 Проєктні рішення з програмного забезпечення програмної системи оцінювання якості

Проєктні рішення щодо програмного забезпечення програмної системи оцінювання якості e-commerce застосунків на Java спрямовані на автоматизацію процесів аналізу метрик програмного коду, побудову регресійних моделей та формування обґрунтованих висновків щодо рівня якості програмних продуктів. У процесі проєктування особливий акцент зроблено на модульності, розширюваності та узгодженості архітектури із розробленими інформаційними моделями.

Програмна система реалізована як багатокомпонентний застосунок, у якому логіка обробки даних відокремлена від логіки представлення результатів. Такий підхід забезпечує можливість подальшого розширення функціональності, зокрема шляхом додавання нових метрик якості, методів нормалізації або альтернативних моделей оцінювання. Архітектура програмної системи базується на фізичній моделі даних, яка визначає структуру збереження метрик RFC, CBO та WMC, результатів нормалізації, перевірки на викиди, параметрів регресійних моделей, а також результатів прогнозування й прийняття рішень щодо якості. У процесі реалізації програмної системи фізична модель даних використовується як основа для організації файлового чи табличного сховища, що гарантує цілісність даних та відтворюваність усіх етапів обчислень. Збереження як первинних, так і нормалізованих даних дозволяє здійснювати порівняльний аналіз результатів до та після застосування нормалізуючих перетворень і перевірки на викиди.

Програмна логіка побудована відповідно до спрощеної моделі класів, у якій виділено основні сутності: модуль попередньої обробки даних, модуль побудови регресійної моделі, модуль розрахунку довірчих інтервалів та інтервалів прогнозування та модуль автоматизованого прийняття рішень щодо якості. Така декомпозиція дозволяє ізолювати математичні обчислення від сервісних операцій, підвищує керованість програмного коду та спрощує

його тестування й супровід. Взаємодія між компонентами реалізується відповідно до моделей їх взаємодії, що визначають послідовність виконання основних операцій оцінювання. Користувач через інтерфейс ініціює процес оцінювання, після чого система здійснює імпорт даних, їх нормалізацію, перевірку на викиди методом квадрата відстані Махаланобіса, побудову нелінійної регресійної моделі та розрахунок показників її якості. На завершальному етапі визначаються довірчі та прогнозні інтервали, після чого формується автоматизоване рішення щодо рівня якості e-commerce застосунку.

3.3.1 Вибір мови програмування та платформи для реалізації програмної системи оцінювання якості

Вибір мови програмування та платформи для реалізації системи оцінювання якості e-commerce застосунків на Java є ключовим етапом проєктування, оскільки він визначає точність відтворення математичної моделі, зручність виконання обчислень, можливості розширення програмного продукту та перспективи його використання у наукових і прикладних дослідженнях. Оскільки основою системи є математична модель, побудована на методах статистики, регресійного аналізу та теорії ймовірностей, головним критерієм вибору мови програмування виступає наявність розвинених засобів чисельних обчислень і моделювання.

Реалізація програми здійснюється мовою Java, що забезпечує переносимість, надійність та можливість інтеграції з іншими інструментами аналізу програмного забезпечення. Запропоновані проєктні рішення створюють цілісну основу для практичного застосування математичної моделі автоматизації прийняття рішень щодо якості e-commerce систем та відкривають перспективи подальшого розвитку програмного засобу.

3.3.2 Фізична модель даних програмної системи оцінювання якості

Фізична модель даних програмної системи оцінювання якості e-commerce застосунків відображає завершену структуру зберігання, обробки та аналізу метрик програмного забезпечення, орієнтовану на автоматизацію процесів прийняття рішень. На рисунку 3.6 наведена фізична модель даних програмної системи оцінювання якості ПС e-commerce.

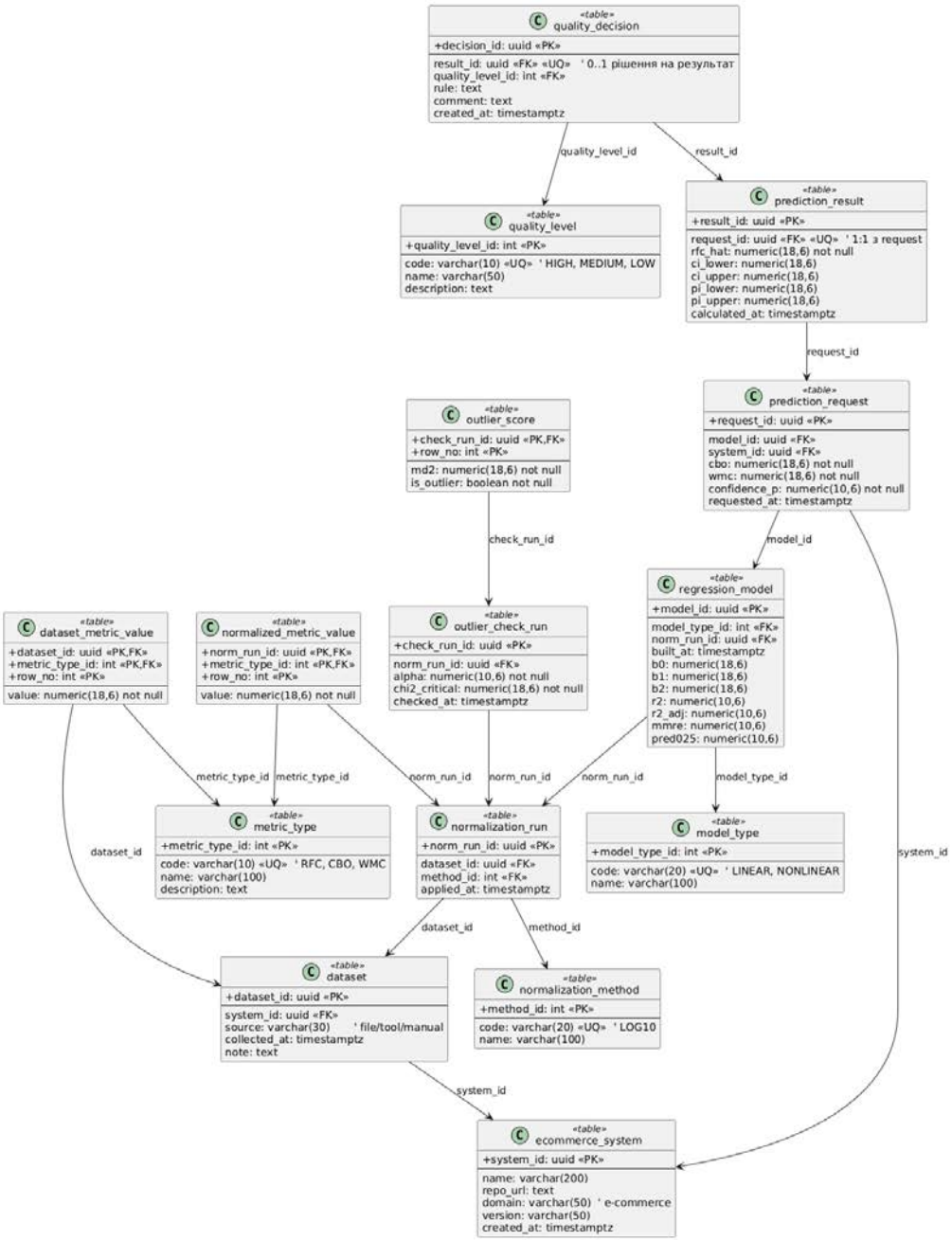

Рисунок 3.6 – Фізична модель даних програмної системи оцінювання якості ПС e-commerce на Java

Центральним елементом моделі виступає сутність ecommerce_system, яка репрезентує програмну систему електронної комерції та містить ідентифікаційні й описові атрибути (назва, домен, репозиторій, версія). Для кожної системи формується набір dataset, що фіксує джерело та момент збору емпіричних даних. Значення метрик RFC, CBO та WMC зберігаються у таблиці dataset_metric_value, де реєструються початкові числові показники.

Попередня обробка даних реалізується через механізм нормалізації. Таблиця normalization_method описує застосовані методи, а таблиця normalization_run фіксує факт використання конкретного методу для певного набору даних. Результати нормалізації зберігаються у normalized_metric_value, що забезпечує відтворюваність експериментів та можливість порівняльного аналізу.

Для виявлення аномальних спостережень у багатовимірному просторі метрик використовується підсистема перевірки викидів. Таблиця outlier_check_run містить параметри перевірки, а таблиця outlier_score – обчислені квадрати відстаней Махаланобіса та ознаку належності спостереження до викидів.

Математичне моделювання реалізується через сутності regression_model та model_type, які зберігають параметри лінійних і нелінійних регресійних моделей, їх статистичні характеристики (R^2, скоригований R^2, MMRE, Pred(0,25)) та часові мітки побудови. На основі обраної моделі формується prediction_request, що містить нормалізовані значення метрик та рівень довіри, а результати прогнозування зберігаються у prediction_result разом із довірчими та прогнозними інтервалами.

Завершальний етап передбачає прийняття управлінського рішення щодо якості програмної системи, яке реалізується через таблицю quality_decision. Вона пов’язує результат прогнозування з довідником quality_level (HIGH, MEDIUM, LOW) та містить текстове пояснення прийнятого рішення.

3.3.3 Модель класів програмної системи автоматизації прийняття рішень щодо якості

На рисунку 3.7 наведена модель класів програмної системи автоматизації прийняття рішень щодо якості ПС e-commerce на Java, що відображає основні програмні компоненти системи, їхні відповідальності та взаємодію в процесі оцінювання якості.

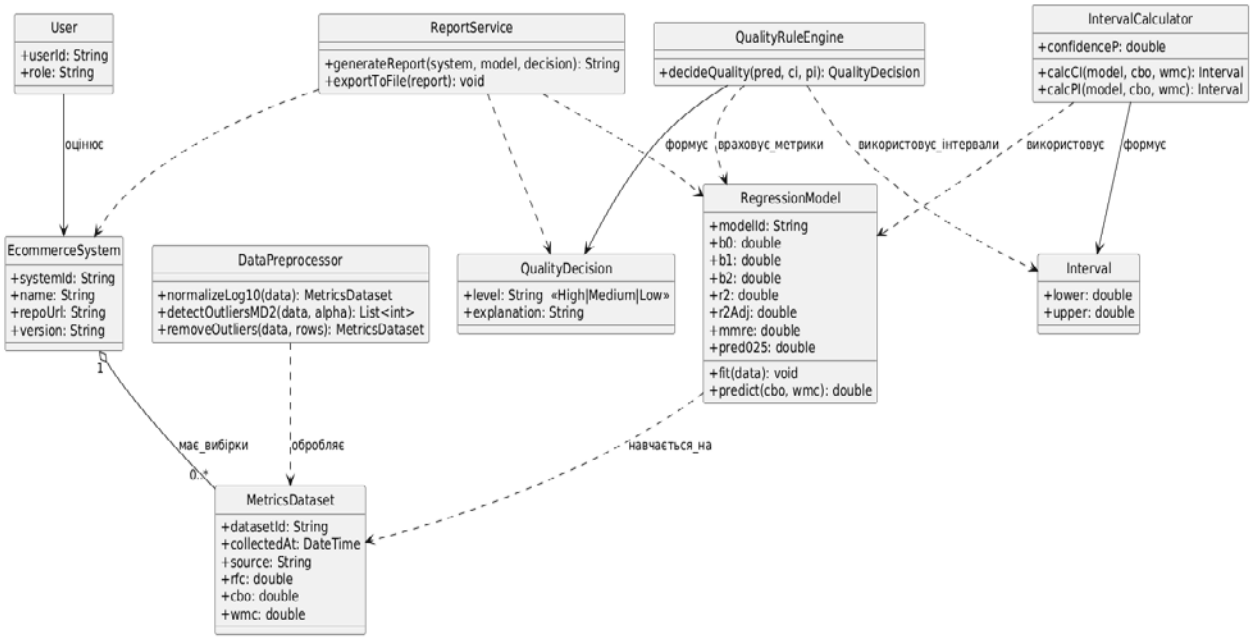

Рисунок 3.7 – Модель класів програмної системи автоматизації прийняття рішень щодо якості ПС e-commerce на Java

Ключовим об’єктом предметної області є клас EcommerceSystem, який описує програмну систему електронної комерції та містить базові ідентифікаційні атрибути (назва, репозиторій, версія). Користувач системи, представлений класом User, ініціює процес оцінювання якості конкретної e-commerce системи.

Збір і підготовка даних реалізовані через клас MetricsDataset, який інкапсулює значення метрик RFC, CBO та WMC разом із метаданими вибірки. Попередня обробка цих даних виконується класом DataPreprocessor, що відповідає за нормалізацію (логарифмічне перетворення), виявлення

багатовимірних викидів за квадратом відстані Махаланобіса та формування скоригованої вибірки для подальшого моделювання.

Математичне моделювання зосереджене у класі RegressionModel, який містить параметри лінійної або нелінійної регресійної моделі, її статистичні показники якості та методи навчання і прогнозування значення метрики RFC на основі факторів CBO та WMC. Для інтервального аналізу результатів прогнозування використовується клас IntervalCalculator, який формує довірчі та прогнозні інтервали, інкапсульовані у класі Interval.

Логіка прийняття управлінського рішення реалізується класом QualityRuleEngine, який на основі прогнозного значення, інтервалів та встановлених правил формує об’єкт QualityDecision. Останній містить дискретну оцінку рівня якості (HIGH, MEDIUM або LOW) та текстове пояснення прийнятого рішення.

Завершальним елементом є сервіс ReportService, що агрегує результати оцінювання, модельні параметри та рішення щодо якості й забезпечує генерацію та експорт аналітичних звітів для користувача.

3.3.4 Моделі взаємодії компонентів програмної системи оцінювання якості

На рисунку 3.8 наведена модель взаємодії компонентів програмної системи оцінювання якості ПС e-commerce на Java, яка відображає послідовність обміну даними між основними функціональними модулями системи в межах сценарію оцінювання якості.

Процес ініціюється користувачем через компонент UI, який забезпечує запуск процедури оцінювання та введення вихідних даних. На першому етапі модуль Data Import виконує завантаження або введення значень метрик RFC, CBO та WMC, після чого первинні дані зберігаються у Data Store та передаються до модуля попередньої обробки.

Компонент Preprocessing відповідає за підготовку даних до моделювання. У його межах здійснюється нормалізація значень метрик із застосуванням десяткового логарифмування, а також перевірка на наявність багатовимірних викидів за допомогою квадрата відстані Махаланобіса. Очищені та нормалізовані дані зберігаються у сховищі та передаються далі для побудови математичної моделі.

На етапі Modeling система виконує побудову нелінійної регресійної моделі оцінювання метрики RFC на основі факторів CBO та WMC. Параметри моделі та статистичні показники її якості фіксуються у сховищі даних. Після цього користувач через інтерфейс вводить значення метрик CBO і WMC для прогнозування, а також рівень довіри, які передаються до наступного модуля.

Компонент Intervals здійснює розрахунок довірчих та прогнозних інтервалів для отриманого прогнозного значення RFC, використовуючи параметри побудованої регресійної моделі. Результати інтервального аналізу передаються разом із прогнозом до модуля прийняття рішень.

У Decision Engine на основі прогнозного значення метрики RFC, інтервалів та заданих правил визначається рівень якості програмної системи. Сформоване рішення передається до модуля Report/Export, який генерує аналітичний звіт і зберігає результати оцінювання у Data Store.

Завершальним етапом є повернення результатів користувачу через UI, де відображаються прогнозне значення RFC, довірчі та прогнозні інтервали, а також підсумкове рішення щодо рівня якості e-commerce програмної системи.

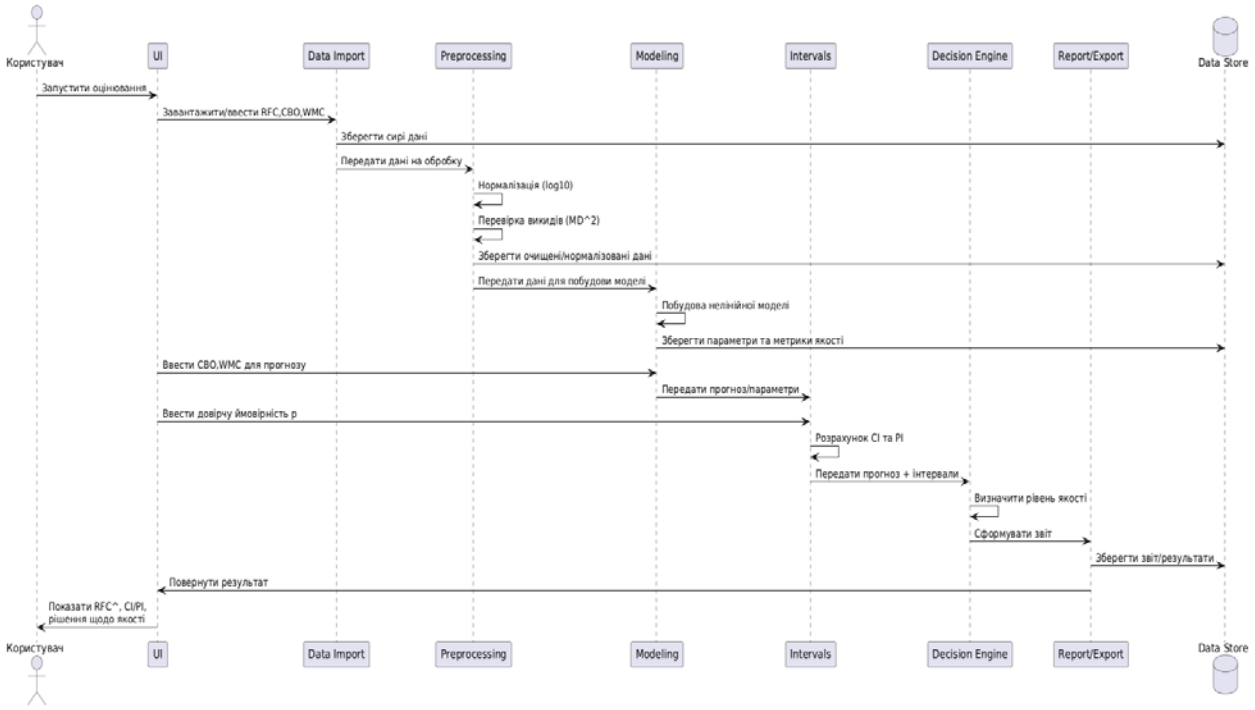

Рисунок 3.8 – Модель взаємодії компонентів програмної системи оцінювання якості ПС e-commerce на Java

Така модель взаємодії забезпечує цілісний та відтворюваний процес автоматизованого оцінювання якості на основі математичного моделювання.

3.3.5 Тестування програмної системи оцінювання якості e-commerce ПС

У межах виконання кваліфікаційної роботи було розроблено програмну систему для оцінювання метрики RFC e-commerce ПС з відкритим кодом на Java, а також здійснено комплексне тестування її функціональності та коректності отриманих результатів. Тестування програмної системи є необхідним етапом життєвого циклу програмного забезпечення, оскільки дозволяє перевірити відповідність програмної реалізації теоретичній математичній моделі, а також оцінити надійність і стабільність роботи системи в різних режимах використання.

На початковому етапі реалізації програмної системи було здійснено кодування алгоритмів, розроблених у межах технічного проєкту, мовою Java

із використанням графічного фреймворку JavaFX, системи збирання Maven та бібліотеки Apache Commons Math для виконання статистичних і матричних обчислень. Повний текст програми наведено в додатку Б.

Загалом результати тестування показали, що програмна система коректно реалізує розроблену математичну модель оцінювання метрики RFC e-commerce ПС на Java, забезпечує стабільну роботу в основних режимах використання та може бути застосована для практичного оцінювання якості програмних систем з відкритим кодом.

3.3.6 Випробування програмної системи автоматизації прийняття рішень щодо якості

Випробування програмної системи автоматизації прийняття рішень щодо якості e-commerce застосунків на Java проводилося з метою перевірки коректності реалізації розробленої математичної моделі та підтвердження відповідності програмної реалізації теоретичним положенням, викладеним у попередніх розділах. Основна увага була зосереджена на перевірці правильності обчислення метрики RFC, формуванні довірчих інтервалів та інтервалів прогнозування, а також на візуальному аналізі емпіричних даних.

Випробування здійснювалося на основі емпіричних даних e-commerce систем з відкритим кодом на Java, що зчитувалися з текстових файлів. Користувач у графічному інтерфейсі програми вводив значення метрик CBO та WMC для досліджуваного застосунку, а також задавав довірчу ймовірність. Після запуску процедури оцінювання система автоматично виконувала всі етапи обробки даних, включаючи застосування нелінійної регресійної моделі та розрахунок інтервальних оцінок.

На рисунку 3.9 наведено приклад роботи графічного інтерфейсу системи в режимі оцінювання якості. Тут показано результати обчислення значення метрики RFC для заданих значень CBO та WMC, а також відповідні

довірчий і прогнозний інтервали при довірчій ймовірності 0,95. Крім того, відображено коефіцієнт детермінації, що характеризує адекватність побудованої нелінійної регресійної моделі. Отримані результати підтверджують коректність реалізації алгоритмів оцінювання та їх узгодженість із математичною моделлю.

Рисунок 3.9 – Графічне вікно програмного засобу з відображеними інтерфейсними елементами

Для аналізу характеру залежності метрики RFC від факторних показників було виконано візуалізацію емпіричних даних. На рисунку 3.10 наведено діаграму розсіювання значень RFC залежно від метрики CBO. З рисунка видно, що між цими показниками спостерігається нелінійна

залежність, а також значна варіативність значень RFC для близьких значень CBO. Це підтверджує доцільність використання регресійного підходу з урахуванням додаткових факторів, зокрема метрики WMC.

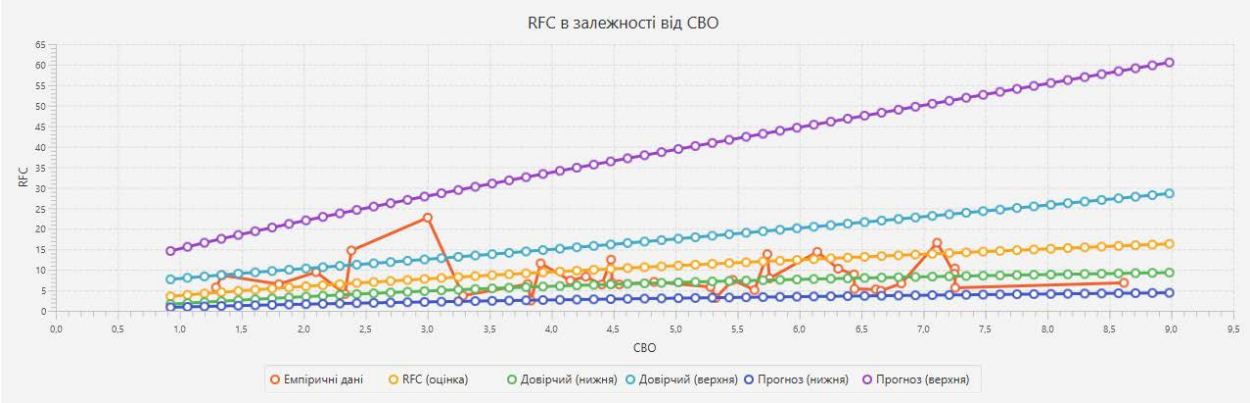

Рисунок 3.10 Емпетричні дані: RFC в залежності від CBO

На рисунку 3.11 представлено діаграму розсіювання значень RFC залежно від метрики WMC. Візуалізація демонструє більш чітку тенденцію зростання значень RFC зі збільшенням WMC, що свідчить про істотний вплив складності класів на кількість відповідей методів. Водночас розподіл точок не є лінійним, що додатково обґрунтовує вибір нелінійної регресійної моделі для оцінювання метрики RFC.

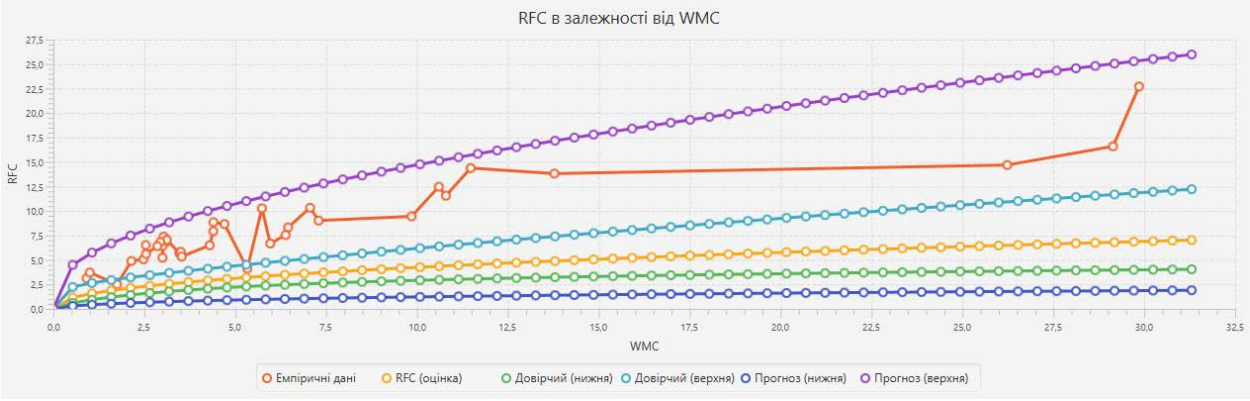

Рисунок 3.11 Емпетричні дані: RFC в залежності від WMC

Узагальнені результати випробувань підтверджують, що програмна система коректно реалізує розроблену математичну модель автоматизації прийняття рішень щодо якості e-commerce застосунків на Java. Отримані числові та графічні результати є логічно узгодженими, а значення метрики RFC та інтервальні прогнози відповідають теоретичним очікуванням. Це засвідчує можливість практичного використання системи для оцінювання якості програмних продуктів з відкритим кодом.

3.4 Висновки за розділом 3

У цьому розділі сформульовано задачу створення програмної системи для оцінювання метрики RFC у ПС електронної комерції з відкритим вихідним кодом, реалізованих мовою Java. У ході роботи було виконано розробку загальносистемних проєктних рішень, проєктних рішень з інформаційного та програмного забезпечення програмної системи, призначеної для автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java на основі метрики RFC.

На стадії розробки загальносистемних проєктних рішень визначено ключові варіанти використання системи оцінювання якості e-commerce застосунків та побудовано відповідні сценарії їх реалізації. Розроблено зовнішню специфікацію програмного засобу для оцінювання метрики RFC у Java-застосунках з відкритим кодом, а також описано стани програми та переходи між ними, що забезпечує формалізацію взаємодії користувача із системою та обробки вхідних даних.

У межах розробки проєктних рішень з інформаційного забезпечення була побудована концептуальна модель даних програмної системи та логічна модель даних програмної системи.

На етапі розробки проєктних рішень з програмного забезпечення було спроєктовано програмний засіб для оцінювання метрики RFC у

Java-орієнтованих e-commerce системах із застосуванням методу покрокової деталізації за принципом «зверху вниз». Розроблено специфікації програмних модулів, визначено їх функціональні ролі та взаємодію між собою, а також побудовано алгоритми роботи модулів, що забезпечують коректний аналіз вихідного коду та обчислення значень метрики RFC для окремих компонентів системи.

Здійснено реалізацію програмного засобу для оцінювання метрики RFC у Java-застосунках з відкритим кодом, проведено комплексне тестування програми та підготовлено відповідну документацію. Крім того, виконано випробування програмного продукту на реальних e-commerce проєктах із відкритим вихідним кодом. Отримані результати оцінювання практично збігаються з даними, наведеними в дослідженнях інших авторів, що підтверджує коректність прийнятих проєктних рішень і працездатність розробленої системи.

4 РОЗРАХУНОК ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ ВІД РОЗРОБКИ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОЇ СИСТЕМИ ДЛЯ ОЦІНЮВАННЯ ЯКОСТІ E-COMMERCE СИСТЕМ НА JAVA

4.1 Розрахунок витрат на створення та впровадження програмної системи

Витрати на створення та експлуатацію програмного забезпечення є важливою складовою економічного обґрунтування доцільності його розробки та впровадження. У процесі розрахунку собівартості програмного продукту враховуються витрати на оплату праці розробника, нарахування на заробітну плату, загальногосподарські витрати, витрати на електроенергію, амортизацію технічних засобів, а також витрати на основні й допоміжні матеріали.

Розробка програмного забезпечення здійснюється одним програмістом. Для розрахунків приймається актуальний на 2025 рік середній місячний оклад працівників у сфері інформаційних технологій в Україні, який становить 67 222 грн. Додаткова заробітна плата приймається у розмірі 20% від основної, що відповідає типовій структурі оплати праці в IT-галузі. Таким чином, основна та додаткова заробітна плата програміста за місяць складає:

- основна заробітна плата – 67 222,00 грн;
- додаткова заробітна плата – 13 444,40 грн;

Отже, загальна сума основної і додаткової заробітної плати становить 80 666,40 грн/міс.

Для виконання робіт із розробки програмного забезпечення використовується персональний комп’ютер у вигляді сучасного бюджетного ноутбука Lenovo IdeaPad 1 15AMN7, вартість якого у 2025 році складає 15 219 грн. Зазначена модель є достатньою за своїми технічними

характеристиками для виконання задач програмування, тестування та налагодження програмних систем.

Витрати на допоміжні матеріали, що використовуються у процесі розробки програмного забезпечення (папір, витратні матеріали для друку, носії інформації та резерв на непередбачені витрати), наведені у таблиці 4.1.

Таблиця 4.1 – Витрати на допоміжні матеріали

Пункти витрат	Сума, грн
Папір (формат А4, 500 аркушів)	224,00
Заправлення картриджа до принтера	300,00
CD-диск	24,00
Непередбачені витрати	54,80
Разом матеріали і комплектуючі	602,80

Загальну вартість розробки програмного забезпечення визначимо за формулою:

$$C_{\text{пр}} = (З_{\text{зп}} + З_{\text{сз}} + З_{\text{зг}} + З_{\text{е}}) \cdot T + З_{\text{м}}, \quad (4.1)$$

де T – тривалість розробки, міс.;

$З_{\text{зп}}$ – основна і додаткова заробітна плата, грн;

$З_{\text{сз}}$ – відрахування на соціальні заходи (єдиний соціальний внесок), грн;

$З_{\text{зг}}$ – загальногосподарські витрати, грн;

$З_{\text{е}}$ – витрати на електроенергію, грн;

$З_{\text{м}}$ – витрати на основні й допоміжні матеріали, грн.

Тривалість розробки програмного забезпечення приймається рівною 1,5 місяця. Відрахування на соціальні заходи у 2025 році для роботодавця становлять 22% від суми основної та додаткової заробітної плати, що відповідає чинному законодавству України. Таким чином:

$$З_{\text{сз}} = 80\,666{,}40 \cdot 0{,}22 = 17\,746{,}61 \text{ грн.}$$

Загальногосподарські витрати приймаються на рівні 10% від основної заробітної плати та становлять:

$$З_{зг} = 67\,222{,}00 \cdot 0{,}10 = 6\,722{,}20 \text{ грн.}$$

Витрати на електроенергію визначаються для непобутового споживача (II клас напруги). Приймається споживана потужність комп'ютера 0,5 кВт, середня тривалість роботи протягом місяця 176 годин (22 робочі дні по 8 годин), а вартість електроенергії для непобутових споживачів у 2025 році – 13,2804 грн за 1 кВт·год з ПДВ. Тоді місячні витрати на електроенергію складуть:

$$З_{е} = 176 \cdot 0{,}5 \cdot 13{,}2804 = 1\,168{,}68 \text{ грн.}$$

Узагальнені витрати на розробку програмного забезпечення наведені у таблиці 4.2.

Таблиця 4.2 – Витрати на розробку програмного забезпечення

Найменування витрат	Одиниця	Значення
Тривалість розробки	міс.	1,5
Основна і додаткова заробітна плата	грн	80 666,40
Відрахування на соціальні заходи	грн	17 746,61
Загальногосподарські витрати	грн	6 722,20
Витрати на матеріали	грн	602,80
Витрати на електроенергію	грн	1 168,68

Підставляючи отримані значення у формулу (4.1), отримаємо загальну вартість розробки програмного забезпечення:

$$C_{пр} = (80\,666{,}40 + 17\,746{,}61 + 6\,722{,}20 + 1\,168{,}68) \cdot 1{,}5 + 602{,}80 = 160\,058{,}62 \text{ грн.}$$

Амортизаційні відрахування на устаткування визначаються у розмірі 60% балансової вартості комп'ютерної техніки на рік:

$$A_{об} = 15\,219 \cdot 0{,}6 = 9\,131{,}40 \text{ грн.}$$

Річні витрати на основні й допоміжні матеріали в масштабах підприємства приймаються на рівні 5% від вартості устаткування та складають:

$$В_м = 15\,219 \cdot 0{,}05 = 760{,}95 \text{ грн.}$$

Річний обсяг роботи персонального комп'ютера у годинах визначається за формулою:

$$Ф_м = 264{,}5 \cdot T_з, \quad (4.2)$$

де $T_з$– середнє місячне завантаження устаткування, яке приймається рівним 4 години, а 264,5 – середня кількість робочих днів у році.

Отже, річний обсяг роботи комп'ютера становить:

$$Ф_м = 264{,}5 \cdot 4 = 1058 \text{ год.}$$

Річні витрати на електроенергію при такому обсязі роботи складуть:

$$З_{е(рік)} = 1058 \cdot 0{,}5 \cdot 13{,}2804 = 7\,025{,}33 \text{ грн.}$$

Таким чином, експлуатаційні витрати на персональний комп'ютер за рік дорівнюють:

$$З_{ер} = 9\,131{,}40 + 760{,}95 + 7\,025{,}33 = 16\,917{,}68 \text{ грн.}$$

Отже, сумарні витрати на створення й експлуатацію програмного забезпечення у перший рік складуть:

$$З_{се} = 160\,058{,}62 + 16\,917{,}68 = 176\,976{,}31 \text{ грн.}$$

4.2 Розрахунок економічної ефективності розробки та впровадження програмної системи

Основним показником економічної ефективності функціонування програмного забезпечення є підвищення ефективності управління інформацією, що проявляється у зниженні витрат на обробку та аналіз даних

при одночасному зростанні швидкості й якості отримання необхідних результатів. Використання автоматизованої системи дозволяє мінімізувати вплив людського фактора, скоротити час виконання рутинних операцій та забезпечити більш стабільну якість обробки інформації.

Крім інших негативних наслідків, ручна обробка інформації спричиняє такі економічні втрати:

- високі витрати на зберігання паперової документації (шафи, сейфи, архівні приміщення тощо);
- підвищені витрати на канцелярські товари та витратні матеріали;
- витрати, пов’язані з виявленням та виправленням раніше допущених помилок, зумовлених людським фактором.

До основних факторів, що визначають приріст економічного ефекту внаслідок упровадження програмного забезпечення, належать:

- підвищення продуктивності праці персоналу;
- вивільнення робочого часу працівників;
- скорочення витрат на супровід і контроль інформаційних процесів.

Крім того, не піддаються прямій грошовій оцінці такі позитивні наслідки впровадження програмного забезпечення, як підвищення оперативності управління, покращення якості прийняття управлінських рішень, зростання надійності результатів, а також удосконалення організації праці в цілому.

Обов’язковою умовою визначення економічної ефективності програмного продукту є порівнянність усіх показників у часі, за цінами та нормативами, що застосовуються для розрахунку витрат, а також за складом і структурою економічних елементів.

Для визначення прямої економічної ефективності приймаємо, що впровадження програмного забезпечення забезпечує вивільнення 0,6 штатної одиниці працівника, що підтверджується експертною оцінкою фахівців

підприємства. Середня місячна заробітна плата працівника у сфері інформаційних технологій у 2025 році приймається рівною 67 222 грн.

Річні витрати на оплату праці 0,6 працівника становлять:

$$\Delta C_{\text{п}} = (67\,222 \cdot 12) \cdot 0{,}6 = 483\,998{,}40 \text{ грн.}$$

З урахуванням річних експлуатаційних витрат на програмне забезпечення та технічні засоби, які відповідно до підрозділу 4.1 складають 16 917,68 грн, вивільнені кошти після впровадження програми становлять:

$$\Delta C_{\text{п}} = 483\,998{,}40 - 16\,917{,}68 = 467\,080{,}72 \text{ грн.}$$

Річний економічний ефект від упровадження програмного забезпечення визначається за формулою:

$$E_p = \Delta C_{\text{п}} - E_{\text{н}} \cdot k, \tag{4.3}$$

де

$\Delta C_{\text{п}}$– вивільнені кошти після впровадження програми, грн;

$E_{\text{н}}$– нормативний коефіцієнт ефективності, який приймається рівним коефіцієнту амортизації (0,6);

k– одноразові витрати на розробку та впровадження програмного забезпечення.

Згідно з підрозділом 4.1, одноразові витрати на створення програмного продукту становлять:

$$k = 160\,058{,}62 \text{ грн.}$$

Тоді величина нормативних відрахувань дорівнює:

$$E_{\text{н}} \cdot k = 0{,}6 \cdot 160\,058{,}62 = 96\,035{,}17 \text{ грн.}$$

Отже, річний економічний ефект від упровадження програмного забезпечення становить:

$$E_p = 467\,080{,}72 - 96\,035{,}17 = 371\,045{,}55 \text{ грн.}$$

Для оцінювання доцільності впровадження програмного забезпечення визначимо строк його окупності, який розраховується за формулою:

$$T = \frac{k}{\Delta C_{\text{п}}}, \tag{4.4}$$

де k– одноразові витрати на розробку програми, грн; $\Delta C_{\text{п}}$– річний економічний ефект до урахування нормативних відрахувань, грн.

Підставляючи числові значення, отримаємо:

$$T = \frac{160\,058{,}62}{467\,080{,}72} = 0{,}34 \text{ року.}$$

Таким чином, строк окупності програмного забезпечення становить приблизно 4,1 місяця, що свідчить про високу економічну ефективність розробки та впровадження даного програмного продукту.

Висновки за розділом 4

У четвертому розділі кваліфікаційної роботи здійснено розширений економічний аналіз, покликаний довести раціональність створення та впровадження програмного забезпечення, що є предметом дослідження. Проведені розрахунки дали змогу не лише визначити повний обсяг витрат на

розробку програмного продукту, а й оцінити очікуваний економічний результат його використання в умовах сучасного підприємства.

Підрозділ 4.1 присвячено формуванню детальної калькуляції витрат на розроблення та подальшу експлуатацію програмного забезпечення з урахуванням економічних показників 2025 року. До кошторису включено основну й додаткову заробітну плату розробника, обов’язкові соціальні відрахування, загальногосподарські витрати, оплату електроенергії для непобутових споживачів, амортизацію комп’ютерної техніки, а також витрати на матеріали. Результати розрахунків засвідчили, що створення програмного продукту потребує 160 058,62 грн, а річні витрати на його експлуатацію становлять 16 917,68 грн. У підсумку загальні витрати першого року дорівнюють 176 976,31 грн – показнику, який відповідає економічним нормам для рішень аналогічного рівня складності.

У підрозділі 4.2 проаналізовано економічні наслідки впровадження розробленого програмного забезпечення. Доведено, що автоматизація інформаційних процесів істотно скорочує витрати, пов’язані з ручною обробкою даних, зменшує ризики, спричинені людським фактором, знижує потребу в паперових носіях та канцелярських матеріалах, а також підвищує продуктивність праці. Використання програмного продукту дозволяє вивільнити 0,6 штатної одиниці, що формує відчутний економічний ефект за рахунок зменшення витрат на оплату праці.

Проведені розрахунки показали, що річний економічний ефект від упровадження програмного забезпечення становить 371 045,55 грн. Період окупності витрат на розробку дорівнює 0,34 року, тобто приблизно 4,1 місяця, що свідчить про високу інвестиційну привабливість та швидку віддачу вкладених коштів. Окрім прямої економічної вигоди, програмний продукт забезпечує низку додаткових переваг: підвищення оперативності управління, покращення якості управлінських рішень, зростання надійності результатів та оптимізацію організації праці.

Отже, результати економічного аналізу переконливо демонструють, що розробка та впровадження програмного забезпечення є фінансово виправданими та економічно ефективними. Отримані показники підтверджують конкурентоспроможність програмного продукту та його здатність підвищувати ефективність діяльності підприємства в реальних умовах.

5 ОХОРОНА ПРАЦІ

Охорона праці становить цілісну систему правових, організаційних та технічних механізмів, спрямованих на забезпечення безпечних умов трудової діяльності. У законодавстві України вона визначається як комплекс заходів, що гарантують збереження життя, здоров’я та працездатності працівників у процесі виконання ними виробничих функцій.

Як інститут трудового права охорона праці охоплює нормативні приписи, які регламентують обов’язки роботодавця та працівника щодо запобігання виробничим ризикам. У системному вимірі це багаторівнева модель управління безпекою праці, що включає правові норми, стандарти, технічні регламенти, санітарно-гігієнічні вимоги та організаційні процедури.

Умови праці, технічний стан обладнання, безпечність технологічних процесів і ефективність засобів захисту мають відповідати вимогам чинних нормативно-правових актів. Особливе значення надається дотриманню санітарно-гігієнічних параметрів, оскільки вони визначають рівень професійного здоров’я та функціональної працездатності персоналу.

Роботодавець зобов’язаний забезпечувати впровадження сучасних технічних і організаційних рішень, спрямованих на мінімізацію виробничих небезпек. До таких рішень належать модернізація робочих місць, використання сертифікованого обладнання, оптимізація трудових процесів та забезпечення працівників засобами індивідуального й колективного захисту.

Невід’ємним елементом системи є навчання та інструктажі з питань охорони праці, пожежної безпеки та реагування на надзвичайні ситуації. Регулярність і послідовність цих заходів формують у працівників стійкі навички безпечної поведінки та знижують імовірність виробничого травматизму.

Порушення вимог охорони праці створює загрозу життю та здоров’ю працівників і може спричинити значні економічні втрати. Законодавство

передбачає дисциплінарну, адміністративну та кримінальну відповідальність за недотримання встановлених норм.

Таким чином, ефективна система охорони праці є ключовою умовою стабільного функціонування підприємства, підвищення продуктивності та збереження трудового потенціалу. Аналіз проблем у цій сфері дає змогу визначити пріоритетні напрями модернізації та обґрунтувати необхідність упровадження інноваційних технічних і організаційних рішень.

5.1 Аналіз шкідливих та небезпечних факторів у офісі комп'ютерної фірми

Робоче середовище офісу комп'ютерної компанії характеризується переважанням інтелектуальної праці, що здійснюється із застосуванням персональних комп'ютерів та офісного обладнання. Попри відсутність технологічних процесів із високим рівнем виробничої небезпеки, офісні умови супроводжуються впливом низки шкідливих і потенційно небезпечних чинників, здатних погіршувати стан здоров'я працівників та знижувати їхню працездатність.

Згідно з класифікацією чинників виробничого середовища, виділяють фізичні, хімічні, біологічні та психофізіологічні фактори. Для офісної діяльності найбільш релевантними є фізичні та психофізіологічні ризики. До фізичних факторів належать параметри мікроклімату, рівень освітленості, шумові та вібраційні впливи, а також електромагнітні поля. Відхилення температури, вологості чи швидкості руху повітря від нормативних значень спричиняє зниження концентрації уваги, втому та погіршення самопочуття. Нестабільний мікроклімат особливо критичний для працівників, які тривалий час працюють за комп'ютером.

Освітлення є ключовим елементом безпечної організації робочого місця. Недостатня або неправильно організована освітленість підвищує

навантаження на зоровий аналізатор, провокуючи зорову втому, головний біль та погіршення гостроти зору. Негативний ефект посилюється контрастом між слабким загальним освітленням і яскравим екраном монітора. Шум і вібрація, що генеруються офісною технікою, хоч і мають невисоку інтенсивність, при тривалому впливі знижують працездатність і сприяють розвитку дратівливості та втоми. Додатковим ризиком є електромагнітні випромінювання, які за перевищення допустимих рівнів можуть негативно впливати на функціонування нервової та серцево-судинної систем.

Психофізіологічні фактори зумовлені специфікою офісної праці: статичним робочим положенням, тривалим напруженням м’язів спини, шиї та верхніх кінцівок, монотонністю операцій і високими вимогами до концентрації уваги. Такі умови сприяють розвитку захворювань опорно-рухового апарату та нервово-емоційного виснаження. Окрему групу становлять ризики, пов’язані з електробезпекою та пожежною безпекою. Порушення правил експлуатації електрообладнання, несправність мереж чи перевантаження електроліній можуть призвести до ураження електричним струмом або виникнення пожежі. Тому необхідним є дотримання вимог електробезпеки та забезпечення офісних приміщень засобами пожежогасіння.

Узагальнений аналіз свідчить, що офіс комп’ютерної фірми є середовищем із комплексом шкідливих і небезпечних факторів, які потребують системного моніторингу та впровадження організаційних, технічних і санітарно-гігієнічних заходів. Зниження їхнього впливу є ключовою умовою формування безпечного робочого простору, збереження здоров’я персоналу та підвищення ефективності професійної діяльності.

5.2 Розрахунок системи пожежогасіння у офісі комп’ютерної фірми

Офіс комп’ютерної компанії належить до адміністративно-побутових приміщень із постійним перебуванням працівників та значною

концентрацією електронного обладнання. Основні джерела пожежної небезпеки пов’язані з експлуатацією комп’ютерів, периферійних пристроїв, мережевої інфраструктури, джерел безперебійного живлення та елементів електромережі. Потенційні загоряння можуть бути спричинені короткими замиканнями, перевантаженням електричних ліній, порушенням правил експлуатації обладнання або помилками персоналу. Тому система первинного пожежогасіння повинна забезпечувати оперативне реагування на початкові осередки пожежі та відповідати нормативним вимогам пожежної безпеки.

Для офісних приміщень характерні загоряння твердих горючих матеріалів, а також пожежі електроустановок, що перебувають під напругою. Це зумовлює необхідність використання як універсальних засобів пожежогасіння, так і спеціалізованих вогнегасників, придатних для гасіння електронного обладнання без ризику його пошкодження.

У розрахунку приймається типовий офіс площею 60 м², розташований на одному поверсі. Відповідно до норм забезпечення вогнегасниками для адміністративних будівель, на поверсі повинно бути встановлено щонайменше два переносні вогнегасники з масою заряду не менше 5 кг. Для цих цілей доцільним є застосування порошкових вогнегасників ВП-5, придатних для гасіння пожеж класів A, B, C та електроустановок до 1000 В.

Окремі вимоги стосуються приміщень із комп’ютерною технікою: нормативи передбачають використання газових вогнегасників із зарядом не менше 3 кг із розрахунку один вогнегасник на кожні 20 м² площі. Кількість таких засобів визначається за формулою:

$$n_{\text{г}} = \left\lceil \frac{S}{20} \right\rceil, \tag{5.1}$$

де S– площа приміщення, м².

При $S = 60$ м^{2}отримаємо:

$$n_{\text{г}} = \left\lceil \frac{60}{20} \right\rceil = 3.$$

Для виконання цієї вимоги обираються вуглекислотні вогнегасники ВВК-3,5, ефективні для гасіння пожеж класу В та електроустановок, які не залишають слідів після застосування та не пошкоджують електронне обладнання.

Таким чином, система первинного пожежогасіння офісу включає два порошкові вогнегасники ВП-5 та три газові вогнегасники ВВК-3,5. Загальна кількість переносних засобів пожежогасіння становить:

$$n_{\Sigma} = 2 + 3 = 5 \text{ одиниць.}$$

Розміщення вогнегасників повинно забезпечувати їхню доступність і не перешкоджати евакуаційним шляхам. Порошкові вогнегасники доцільно встановлювати біля виходів і коридорів, тоді як газові – у зонах розташування комп’ютерної та мережевої техніки. Такий склад і конфігурація засобів пожежогасіння забезпечують відповідність офісу вимогам пожежної безпеки та створюють умови для швидкої локалізації можливих загорянь.

5.3 Розробка заходів щодо зменшення впливу шкідливих та небезпечних факторів

З метою запобігання або зменшення впливу шкідливих і небезпечних виробничих факторів під час організації робочих місць та робочої зони в офісі комп’ютерної фірми необхідно дотримуватися загальних вимог, установлених законодавством України з охорони праці. Реалізація комплексу організаційних, технічних, санітарно-гігієнічних і профілактичних заходів спрямована на створення безпечних умов праці, збереження здоров’я працівників і підвищення ефективності їхньої професійної діяльності.

Відповідно до Закону України «Про охорону праці», умови праці на робочому місці, безпека технологічних процесів, устаткування та інших засобів виробництва, стан засобів колективного й індивідуального захисту, що використовуються працівниками, а також санітарно-побутові умови

повинні відповідати вимогам чинних нормативно-правових актів. Роботодавець зобов'язаний забезпечити належні умови праці в кожному структурному підрозділі підприємства та несе відповідальність за їх дотримання.

Особливу увагу в офісі комп'ютерної фірми слід приділяти організації робочого місця працівника, який тривалий час працює за персональним комп'ютером.

Як відомо, тривала робота за комп'ютером та з документами за умов недостатнього або неправильно організованого освітлення може призвести до значного перенапруження зорового апарату, розвитку головного болю, швидкої втоми та зниження працездатності. У зв'язку з цим вимоги до освітлення робочих місць є особливо важливими. Вимоги до природного та штучного освітлення приміщень регламентуються державними будівельними нормами ДБН В.2.5-28-2006 «Природне і штучне освітлення».

Робочі місця в офісі необхідно організовувати таким чином, щоб природне світло падало з лівого боку відносно працівника, що відповідає вимогам ДСанПіН 3.3.2.007-98. При цьому розташування робочих столів і моніторів повинно виключати потрапляння прямого світла в очі працівника та появу відблисків на екрані. Для зменшення впливу шкідливих факторів зорового навантаження доцільно застосовувати екранні фільтри, локальні світлофільтри та інші засоби індивідуального захисту органів зору, які пройшли відповідні випробування та мають чинний гігієнічний сертифікат.

Штучне освітлення офісних приміщень повинно здійснюватися системою загального рівномірного освітлення, що забезпечує нормативні значення освітленості в робочій зоні. У приміщеннях, де переважає робота з документами, допускається застосування комбінованого освітлення, тобто поєднання загального освітлення з місцевими світильниками. Як джерела штучного освітлення доцільно використовувати люмінесцентні лампи, які забезпечують рівномірний розподіл світлового потоку та знижують навантаження на зір. Світильники загального освітлення рекомендується

розміщувати у вигляді суцільних або переривчастих ліній, розташованих збоку від робочих місць, зазвичай ліворуч і паралельно лінії зору працівників.

Допускається застосування ламп розжарювання у світильниках місцевого освітлення, а також використання металогалогенних ламп потужністю до 250 Вт у разі організації відбитого освітлення в адміністративно-громадських приміщеннях. При цьому коефіцієнт пульсації освітлення не повинен перевищувати 5%, що відповідає санітарно-гігієнічним вимогам. Рівень освітленості на робочому столі в зоні розміщення документів має перебувати в межах 300–500 лк, а освітленість екрана монітора не повинна перевищувати 300 лк. Світильники місцевого освітлення необхідно встановлювати таким чином, щоб вони не створювали відблисків на поверхні екрана.

Для підтримання нормативних значень освітленості в приміщеннях офісу необхідно здійснювати регулярне обслуговування освітлювальних приладів. Зокрема, відповідно до вимог ДСанПіН 3.3.2.007-98, вікна та світильники слід очищувати від забруднень не рідше двох разів на рік, а лампи, що перегоріли або втратили світлові характеристики, повинні своєчасно замінюватися.

6 ОХОРОНА НАВКОЛИШНЬОГО СЕРЕДОВИЩА

6.1 Вступ

У сучасних умовах інтенсивного техногенного розвитку питання охорони довкілля стають ключовими для забезпечення сталого економічного та соціального прогресу. Зростання масштабів виробництва, активне впровадження інформаційних технологій і підвищення енергоспоживання посилюють антропогенний тиск на природні системи, що вимагає впровадження екологічно орієнтованих управлінських та технічних рішень.

Охорона навколишнього середовища передбачає комплекс заходів, спрямованих на раціональне використання природних ресурсів, запобігання їх деградації та мінімізацію негативних наслідків господарської діяльності. У сфері інформаційних технологій ці питання набувають особливої ваги через використання комп’ютерної техніки, значне енергоспоживання та формування електронних відходів.

Хоча розробка програмного забезпечення не створює прямих промислових викидів, її екологічний вплив проявляється опосередковано: теплові викиди, електромагнітні поля, споживання електроенергії та накопичення відпрацьованої техніки. Тому забезпечення екологічної безпеки в діяльності комп’ютерних фірм потребує системного аналізу та впровадження відповідних заходів.

Метою даного розділу є визначення основних екологічних проблем, пов’язаних із функціонуванням комп’ютерної фірми, аналіз чинників негативного впливу на довкілля та обґрунтування необхідності впровадження заходів, спрямованих на його зменшення. Такий підхід дає змогу оцінити екологічну ефективність використання комп’ютерної техніки та забезпечити відповідність діяльності підприємства вимогам екологічного законодавства України.

6.2 Аналіз існуючої проблеми необхідності охорони навколишнього середовища

Охорона довкілля та раціональне використання природних ресурсів є ключовими умовами сталого розвитку України. Зростання техногенного навантаження підсилює значення екологічної безпеки та потребує комплексного регулювання на всіх рівнях управління. Нормативну основу екологічної політики формує Закон України «Про охорону навколишнього природного середовища» та спеціальне природоохоронне законодавство, яке визначає принципи збереження природних ресурсів, запобігання забрудненню та забезпечення екологічної безпеки населення.

Сучасні техногенні процеси – промисловість, транспорт, ІТ-інфраструктура є супроводжуються зростанням енергоспоживання, утворенням відходів і накопиченням забруднювальних речовин у біосфері. Це спричиняє деградацію екосистем, втрату біорізноманіття та погіршення стану здоров’я населення. У сфері розробки програмного забезпечення екологічний вплив пов’язаний із роботою комп’ютерної техніки та мережевого обладнання: споживанням електроенергії, тепловими викидами, електромагнітними полями та зростанням обсягів електронних відходів.

Недотримання вимог енергоефективності та правил експлуатації техніки призводить до підвищеного навантаження на довкілля та скорочення ресурсу обладнання.

Отже, зменшення екологічного впливу діяльності комп’ютерних фірм потребує впровадження енергоощадних технологій, дотримання екологічних стандартів та організаційних заходів, спрямованих на раціональне використання ресурсів і безпечне поводження з електронними відходами.

6.3 Забруднення навколишнього середовища співробітниками офісу комп’ютерної фірми

Діяльність офісних працівників комп’ютерної фірми, хоча й не пов’язана з прямими промисловими викидами, супроводжується певним антропогенним навантаженням на довкілля. Основними його джерелами є експлуатація комп’ютерної техніки, споживання електроенергії, використання офісних матеріалів та формування різних видів відходів.

Найвагомішим чинником опосередкованого забруднення є енергоспоживання, пов’язане з роботою комп’ютерів, серверів, периферійних пристроїв, систем кондиціонування та освітлення. Оскільки виробництво електроенергії, особливо на теплових електростанціях, супроводжується викидами парникових газів, нераціональне використання енергії в офісі сприяє погіршенню екологічного стану. Додаткове навантаження формують тверді побутові та офісні відходи – папір, упаковка, пластик, використані картриджі, батарейки та дрібні електронні компоненти. За відсутності сортування та належної утилізації вони накопичуються на полігонах, де можуть забруднювати ґрунти та водні ресурси.

Окрему загрозу становлять електронні відходи, що утворюються внаслідок зношування комп’ютерної техніки. Такі пристрої містять токсичні речовини, які за неналежної утилізації здатні завдавати шкоди екосистемам і здоров’ю людини. Недотримання екологічних вимог під час оновлення обладнання підсилює цей негативний вплив.

Використання офісних витратних матеріалів також має екологічні наслідки. Надмірне споживання паперу збільшує потребу у вирубці лісів, витратах води та енергії на його виробництво, а також сприяє накопиченню паперових відходів.

6.4 Розробка заходів щодо зменшення забруднення

Використання комп’ютерної техніки в офісі комп’ютерної фірми супроводжується утворенням електронних та електротехнічних відходів, що містять метали, пластмаси, скло та низку небезпечних компонентів. Неналежне поводження з такими матеріалами створює ризики для довкілля, оскільки вони можуть містити токсичні речовини та цінні метали, вилучення яких потребує спеціалізованих технологій.

Утилізація застарілої техніки передбачає її розбирання на окремі фракції – метали, пластмаси, електронні плати, кабелі, акумулятори та скляні елементи. Хоча повторне використання деталей у первинному вигляді зазвичай недоцільне, більшість компонентів може бути перероблена як вторинна сировина, що сприяє зменшенню обсягів відходів і раціональному використанню ресурсів. Особливо цінними є електронні плати, які містять дорогоцінні та рідкоземельні метали, однак їх переробка можлива лише на спеціалізованих підприємствах.

Офісні структури не здійснюють самостійної утилізації електронних відходів, тому оптимальним є їх облік, тимчасове зберігання та передача ліцензованим організаціям. Це забезпечує дотримання екологічних вимог і мінімізує ризики забруднення.

Під час експлуатації комп’ютерів також витрачаються супутні ресурси електроенергія, папір, картриджі та інші витратні матеріали. Екологічно доцільними є енергоощадні режими роботи техніки, двосторонній друк, перехід на електронний документообіг і залучення паперу до вторинної переробки. Відновлення картриджів дозволяє багаторазово їх використовувати, зменшуючи кількість відходів.

Сучасна комп’ютерна техніка має тривалий строк служби, що знижує частоту утворення відходів. У сервісних центрах доцільно організовувати збір відпрацьованих компонентів, що містять цінні метали, з подальшою передачею їх на переробку.

Таким чином, система поводження з електронними відходами в офісі повинна базуватися на принципах ресурсоефективності, мінімізації утворення відходів та дотримання екологічних норм, що відповідає вимогам сталого розвитку.

ВИСНОВКИ

У кваліфікаційній роботі розв’язана практична задача, спрямована на автоматизацію процесу прийняття рішень щодо якості e-commerce програмних систем з відкритим вихідним кодом, розроблених мовою Java, на основі метрик об’єктно-орієнтованого дизайну RFC, CBO та WMC.

У процесі дослідження було проаналізовано наявні підходи та регресійні моделі оцінювання якості програмних систем, зокрема тих, що застосовуються у сфері електронної комерції. Проведений аналіз засвідчив, що традиційні лінійні моделі не завжди забезпечують коректність оцінювання через порушення припущень щодо нормальності розподілу залишків, що підтвердило доцільність використання нелінійних регресійних залежностей.

Було виконано підбір та формування вибірки даних для e-commerce систем з відкритим кодом на Java. Для підвищення достовірності результатів проведено перевірку багатовимірних даних на наявність викидів із застосуванням методів математичної статистики та нормалізуючих перетворень. У результаті сформовано скориговану вибірку, придатну для побудови регресійної моделі.

У роботі вдосконалено двофакторну нелінійну регресійну модель оцінювання метрики RFC залежно від показників CBO та WMC шляхом застосування нормалізуючого перетворення у вигляді десяткового логарифмування. Запропонована модель дала змогу врахувати негаусовий характер розподілу метрик та забезпечити більш коректне статистичне оцінювання якості e-commerce програмних систем на Java.

На основі побудованої нелінійної регресійної моделі створено програмну систему для автоматизації прийняття рішень щодо якості e-commerce застосунків на Java. Розроблена програмна система підтримує інтервальне оцінювання та може використовуватися як інструмент для

аналізу архітектурної складності, рівня зв’язності компонентів і функціонального навантаження e-commerce застосунків на Java.

Реалізацію програмної системи виконано мовою програмування Java із використанням графічного фреймворку JavaFX, системи збирання Maven та бібліотеки Apache Commons Math для виконання статистичних і матричних обчислень.

Програмна система забезпечує обробку емпіричних даних метрик RFC, CBO та WMC, побудову нелінійної регресійної моделі у логарифмічному просторі, а також розрахунок довірчих і прогнозних інтервалів відповідно до заданого рівня довірчої ймовірності.

Розроблено повний комплект програмної документації, зокрема технічне завдання, текст та опис програмної системи, інструкцію користувача, а також програму і методику випробувань, наведені у додатках А, Б, В, Г, Д. Проведені випробування підтвердили працездатність розробленої програмної системи та її придатність для використання в задачах автоматизованого аналізу й управління якістю програмних систем електронної комерції, розроблених мовою Java.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Prykhodko S., Prykhodko N. A technique for detecting software quality based on the confidence and prediction intervals of nonlinear regression for RFC metric // Proceedings of the 17th IEEE International Conference on Computer Sciences and Information Technologies (CSIT 2022). Україна, 2022. С. 499–502. DOI: 10.1109/CSIT56992.2022.10086061.

2. Prykhodko S. Evaluating quality of software systems by the confidence and prediction intervals of nonlinear regressions for RFC, CBO, and WMC metrics // WSEAS Transactions on Systems. 2024. Т. 23. С. 93–101. DOI: 10.37394/23202.2024.23.12.

3. Prykhodko S. Statistical analysis of the three-dimensional data of RFC, CBO, and WMC metrics for the software developed in Java // Proceedings of the International Conference on Artificial Intelligence and Information Technologies (ICAIIT 2025). 2025. С. 39–44. DOI: 10.13142/kt10006.168.

4. Prykhodko S. A nonlinear regression model for early LOC estimation of Java-based apps // WSEAS Transactions on Systems. 2024. Т. 23. С. 43–51. DOI: 10.37394/23202.2024.23.6.

5. Малюк А.С., Надточий Т.М., Макарова Л.М. Математична модель для автоматизації прийняття рішень щодо якості програмних систем e-commerce, що створюються мовою Java. Матеріали VIII Всеукраїнської науково-практичної інтернет-конференції студентів, аспірантів та молодих вчених за тематикою «Сучасні комп’ютерні системи та мережі в управлінні» (24 листопада 2025 р., м. Херсон, м. Хмельницький) / за ред. А.А. Григорової. Херсон: Книжкове видавництво ФОП Вишемирський В.С., 2025. С. 162-163.

6. TIOBE Software. TIOBE Index for November 2025. URL: https://www.tiobe.com/ (дата звернення: 23.12.2025).

7. Deitel P., Deitel H. Java: How to Program. 11th ed. Pearson, 2022. 1536 p.

8. Oracle. Java Platform, Standard Edition Documentation. URL: https://docs.oracle.com/javase/ (дата звернення: 23.12.2025).

9. Richardson C. Microservices Patterns: With Examples in Java. Manning Publications, 2021. 520 p.

10. Sharma A., Gupta R. Quality assurance metrics and performance evaluation in Java-based enterprise systems // Journal of Software Engineering and Applications. 2024. Т. 17, № 3. С. 112–128.

11. Box G. E. P., Cox D. R. An Analysis of Transformations // Journal of the Royal Statistical Society. Series B. 1964.

12. Johnson R. A., Wichern D. W. Applied Multivariate Statistical Analysis. Pearson, 2007. 800 p.

13. Chidamber S. R., Kemerer C. F. A Metrics Suite for Object-Oriented Design // IEEE Transactions on Software Engineering. 1994.

14. Barnett V., Lewis T. Outliers in Statistical Data. Wiley, 1994. 608 p.

15. Rousseeuw P. J., Leroy A. M. Robust Regression and Outlier Detection. Wiley, 1987. 329 p.

16. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning. Springer, 2009. 745 p.

17. Mahalanobis P. C. On the generalized distance in statistics // Proceedings of the National Institute of Sciences of India. 1936.

18. ISO/IEC 25010. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models. URL: https://iso25000.com/en/iso-25000-standards/iso-25010 (дата звернення: 23.12.2025).

19. ISO. ISO/IEC 25010:2011 – Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE). URL: https://www.iso.org/standard/35733.html (дата звернення: 23.12.2025).

20. Bansiya J., Davis C. A Hierarchical Model for Object-Oriented Design Quality Assessment (QMOOD). URL: https://pdfs.semanticscholar.org/37a1/b6b861d042dbacf6c7eaa2927d3dfed2043c.pdf (дата звернення: 23.12.2025).

21. Capretz L. F. An Empirical Validation of Object-Oriented Design Metrics. URL: https://thescipub.com/pdf/jcssp.2008.571.577.pdf (дата звернення: 23.12.2025).

22. GitHub. Офіційний вебсайт платформи для хостингу та спільної розробки програмного забезпечення. URL: https://github.com/ (дата звернення: 23.12.2025).

23. Aniche M. CK: Tool for calculating Chidamber and Kemerer (CK) metrics for Java projects. Репозиторій GitHub. URL: https://github.com/mauricioaniche/ck (дата звернення: 23.12.2025).

ДОДАТОК А – Технічне завдання на розробку програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java

Вступ

Назва розробки – «Програмна система автоматизації прийняття рішень щодо якості програмних систем e-commerce на Java (JavaFX)», надалі – *програма*.

Підстави для розробки

Програма розробляється в межах виконання кваліфікаційної роботи та спрямована на практичну реалізацію математичної моделі оцінювання якості програмних систем електронної комерції з відкритим кодом, розроблених мовою Java.

Призначення розробки

Програма призначена для автоматизованого оцінювання якості програмних систем e-commerce з відкритим кодом на Java на основі метрик програмного забезпечення RFC, CBO та WMC із використанням нелінійної регресійної моделі, реалізованої засобами мови Java у середовищі JavaFX.

Програмний засіб забезпечує підтримку прийняття рішень щодо якості програмних систем шляхом аналізу емпіричних даних та побудови довірчих і прогнозних інтервалів для метрики RFC.

Вимоги до програми:

Вимоги до функціональних характеристик

Вимоги до складу функцій, що виконуються.

Програма повинна забезпечувати виконання таких основних функцій:

- зчитування з текстового файлу емпіричних даних метрик RFC, CBO та WMC програмних систем e-commerce з відкритим кодом на Java;

- введення через графічний інтерфейс значень метрик CBO та WMC програмної системи, для якої здійснюється оцінювання метрики RFC;
- введення користувачем значення довірчої ймовірності для побудови довірчих та прогнозних інтервалів;
- нормалізацію емпіричних даних із використанням десяткового логарифмічного перетворення;
- побудову нелінійної регресійної моделі оцінювання метрики RFC залежно від метрик CBO та WMC;
- обчислення статистичних показників якості моделі, зокрема коефіцієнта детермінації R^2 та середньої відносної похибки MRE (у разі наявності фактичного значення RFC);
- побудову довірчого інтервалу та інтервалу прогнозування для метрики RFC;
- візуалізацію результатів оцінювання у вигляді графіків залежності RFC від CBO та WMC;
- виведення результатів оцінювання на екран, включаючи прогнозоване значення RFC, межі інтервалів та показники якості моделі.
-

Вимоги до організації вхідних та вихідних даних

Вхідні дані:

- емпіричні дані метрик RFC, CBO та WMC, подані у текстовому файлі;
- значення метрик CBO та WMC програмної системи, що оцінюється;
- значення довірчої ймовірності;
- (за потреби) фактичне значення метрики RFC для обчислення MRE.

Вихідні дані:

- прогнозоване значення метрики RFC;
- межі довірчого інтервалу та інтервалу прогнозування;

- значення коефіцієнта детермінації R^2;
- значення середньої відносної похибки MRE;
- графічні залежності RFC від CBO та WMC;
- сформоване рішення щодо якості програмної системи e-commerce.
-

Вимоги до надійності

Спеціальні вимоги до надійності не висуваються.

Вимоги до умов експлуатації

Спеціальні вимоги до умов експлуатації не висуваються.

Вимоги до складу та параметрів технічних засобів

Програма повинна експлуатуватися на обчислювальних засобах із такими мінімальними характеристиками:

- процесор рівня Intel Core i3 або еквівалентний;
- оперативна пам’ять не менше 2 ГБ;
- вільний простір на жорсткому диску не менше 600 МБ.

Вимоги до інформаційної та програмної сумісності

Для запуску та коректної роботи програми необхідні:

- операційна система Windows 10 / 11 (64-бітна);
- Java Development Kit (JDK) версії 17 або вище;
- система збирання проєктів Apache Maven;
- середовище розробки або запуску JavaFX.
-

Техніко-економічні показники

Техніко-економічні показники в межах даної роботи не розраховуються.

Стадії та етапи розробки

Стадії та етапи розробки програмного забезпечення визначаються відповідно до вимог кваліфікаційної роботи та наведені в таблиці А.1.

Таблиця А.1 - Стадії та етапи розробки

Стадія розробки	Етапи робіт	Термін виконання
1. Технічне завдання	Розробка та затвердження технічного завдання	14.09.2025
2. Ескізний проект	Розробка ескізного проекту	23.09.2025
	Затвердження ескізного проекту	25.09.2025
3. Технічних проект	Розробка технічного проекту	07.10.2025
	Затвердження технічного проекту	11.10.2025
4. Робочий проекст	Розробка програми	20.10.2025
	Розробка програмної документації	22.10.2025
	Випробування програми	17.11.2025

Порядок контролю і приймання

Контроль та приймання програмного забезпечення здійснюються уповноваженими представниками замовника за участю представника розробника.

Для проведення процедури приймання замовнику передається комплект програмної документації, підготовлений відповідно до вимог технічного завдання. Представники замовника здійснюють перевірку програмного забезпечення на відповідність установленим у технічному завданні вимогам.

За підсумками проведеної перевірки оформлюється акт приймання, який засвідчує виконання робіт та підтверджує відповідність програмного продукту визначеним вимогам.

ДОДАТОК Б – Код програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java

MainApp.java:

```
package ua.rfcapp;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.chart.LineChart;
import javafx.scene.chart.NumberAxis;
import javafx.scene.chart.XYChart;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.stage.FileChooser;
import javafx.stage.Stage;

import java.io.File;
import java.nio.file.Path;
import java.text.DecimalFormat;

public class MainApp extends Application {

    private static final DecimalFormat DF4 = new
DecimalFormat("0.0000");

    private RfcModel.Dataset dataset;

    @Override
    public void start(Stage stage) {
        // File controls
        TextField fileField = new TextField();
        fileField.setPromptText("Вкажіть файл RFC CBO WMC
(36 рядків)");
        fileField.setPrefColumnCount(28);

        Button browseBtn = new Button("Обрати...");
        Button loadBtn = new Button("Ввести");
        // Inputs
        TextField cboField = new TextField("8.625");
        TextField wmcField = new TextField("22.25");
        TextField pConfField = new TextField("0.95");
```

```
        // Optional actual RFC
        CheckBox hasActual = new CheckBox("є (включити)");
        hasActual.setSelected(false);
        TextField actualRfcField = new TextField("0");

actualRfcField.disableProperty().bind(hasActual.selectedPro
perty().not());

        // Dataset info labels
        Label nLabel = new Label("-");
        Label sigmaLabel = new Label("-");
        Label r2Label = new Label("-");

        // Results labels
        Label rfcLabel = new Label("-");
        Label confLowLabel = new Label("-");
        Label confHighLabel = new Label("-");
        Label predLowLabel = new Label("-");
        Label predHighLabel = new Label("-");
        Label mreLabel = new Label("-");

        LineChart<Number, Number> chartCbo =
createChart("RFC в залежності від CBO", "CBO", "RFC");
        LineChart<Number, Number> chartWmc =
createChart("RFC в залежності від WMC", "WMC", "RFC");

        Button evalBtn = new Button("Оцінювання");
        evalBtn.setDisable(true);

        browseBtn.setOnAction(e -> {
            FileChooser fc = new FileChooser();
            fc.setTitle("Оберіть файл з даними (RFC CBO
WMC)");
            File f = fc.showOpenDialog(stage);
            if (f != null)
fileField.setText(f.getAbsolutePath());
        });
        loadBtn.setOnAction(e -> {
            try {
                String p = fileField.getText().trim();
                if (p.isEmpty()) throw new
IllegalArgumentException("Вкажіть шлях до файлу.");
                Path path = Path.of(p);

                dataset = RfcModel.loadDataset(path);
```

```
nLabel.setText(String.valueOf(dataset.n()));

sigmaLabel.setText(DF4.format(dataset.sigmaLog()));
                r2Label.setText(DF4.format(dataset.r2()));

                plotAll(chartCbo, chartWmc);

                evalBtn.setDisable(false);

                info("Дані завантажено: n=" + dataset.n());
            } catch (Exception ex) {
                dataset = null;
                evalBtn.setDisable(true);
                showError("Помилка завантаження",
ex.getMessage());
            }
        });

        // Evaluate
        evalBtn.setOnAction(e -> {
            try {
                if (dataset == null) throw new
IllegalStateException("Спочатку натисніть 'Ввести' для
завантаження даних.");

                double cbo = parseDouble(cboField, "CBO");
                double wmc = parseDouble(wmcField, "WMC");
                double pConf = parseDouble(pConfField,
"Довірча ймовірність");

var res = RfcModel.evaluate(dataset, cbo, wmc, pConf);

rfcLabel.setText(DF4.format(res.rfc()));

confLowLabel.setText(DF4.format(res.confLow()));

confHighLabel.setText(DF4.format(res.confHigh()));

predLowLabel.setText(DF4.format(res.predLow()));

predHighLabel.setText(DF4.format(res.predHigh()));

r2Label.setText(DF4.format(res.r2()));

if (hasActual.isSelected()) {
```

```
double actual = parseDouble(actualRfcField, "Фактичний
RFC");
                    if (actual == 0) throw new
IllegalArgumentException("Фактичний RFC не може бути 0.");
                    double mre = Math.abs((actual -
res.rfc()) / actual);
                    mreLabel.setText(DF4.format(mre));
                } else {
                    mreLabel.setText("-");
                }
                plotAll(chartCbo, chartWmc);

            } catch (Exception ex) {
                showError("Помилка оцінювання",
ex.getMessage());
            }
        });

        Button closeBtn = new Button("Закрити");
        closeBtn.setOnAction(e -> stage.close());

        // Layout
        GridPane form = new GridPane();
        form.setHgap(10);
        form.setVgap(8);

        int r = 0;

        form.add(new Label("Файл з даними (RFC CBO WMC)"),
0, r);
        HBox fileBox = new HBox(8, fileField, browseBtn,
loadBtn);
        fileBox.setAlignment(Pos.CENTER_LEFT);
        form.add(fileBox, 1, r++, 2, 1);

        form.add(new Label("Значення CBO на рівні
застосунку"), 0, r);
        form.add(cboField, 1, r++);

        form.add(new Label("Значення WMC на рівні
застосунку"), 0, r);
        form.add(wmcField, 1, r++);

        form.add(new Label("Довірча ймовірність"), 0, r);
        form.add(pConfField, 1, r++);
```

```
        HBox actualBox = new HBox(10, hasActual, new
Label("RFC факт:"), actualRfcField);
        actualBox.setAlignment(Pos.CENTER_LEFT);
        form.add(actualBox, 0, r, 2, 1);
        r++;

        GridPane dsInfo = new GridPane();
        dsInfo.setHgap(10);
        dsInfo.setVgap(6);
        dsInfo.add(new Label("n:"), 0, 0);
        dsInfo.add(nLabel, 1, 0);
        dsInfo.add(new Label("σ(log10):"), 0, 1);
        dsInfo.add(sigmaLabel, 1, 1);
        dsInfo.add(new Label("R²:"), 0, 2);
        dsInfo.add(r2Label, 1, 2);

        HBox buttons = new HBox(10, evalBtn, closeBtn);
        buttons.setAlignment(Pos.CENTER_LEFT);

        GridPane results = new GridPane();
        results.setHgap(10);
        results.setVgap(8);

        int rr = 0;
        results.add(new Label("Результат нел. регресії
метрики RFC"), 0, rr);
        results.add(rfcLabel, 1, rr++);

        results.add(new Label("Нижня границя довір.
інтервалу"), 0, rr);
        results.add(confLowLabel, 1, rr++);

        results.add(new Label("Верхня границя довір.
інтервалу"), 0, rr);
        results.add(confHighLabel, 1, rr++);

        results.add(new Label("Нижня границя інтервалу
прогноз."), 0, rr);
        results.add(predLowLabel, 1, rr++);

        results.add(new Label("Верхня границя інтервалу
прогноз."), 0, rr);
        results.add(predHighLabel, 1, rr++);

        results.add(new Label("MRE (якщо введено факт
RFC)"), 0, rr);
```

```
        results.add(mreLabel, 1, rr++);

        VBox left = new VBox(12,
                titled("Дані для оцінювання", form),
                titled("Параметри з файлу", dsInfo),
                buttons,
                titled("Результати оцінювання", results)
        );
        left.setPadding(new Insets(10));
        left.setPrefWidth(560);

        VBox charts = new VBox(10, chartCbo, chartWmc);
        charts.setPadding(new Insets(10));
        charts.setPrefWidth(760);

        BorderPane root = new BorderPane();
        root.setLeft(left);
        root.setCenter(charts);

        Scene scene = new Scene(root, 1360, 780);
        stage.setTitle("Оцінювання метрики RFC — JavaFX (з
36 даними)");
        stage.setScene(scene);
        stage.show();
    }

    private void plotAll(LineChart<Number, Number>
chartCbo, LineChart<Number, Number> chartWmc) {
        chartCbo.getData().clear();
        chartWmc.getData().clear();

        if (dataset == null) return;

        // 1) Эмпирические точки как в Scilab
        XYChart.Series<Number, Number> empCbo = new
XYChart.Series<>();
        empCbo.setName("Емпіричні дані");

        XYChart.Series<Number, Number> empWmc = new
XYChart.Series<>();
        empWmc.setName("Емпіричні дані");

        double minCbo = Double.POSITIVE_INFINITY, maxCbo =
Double.NEGATIVE_INFINITY;
        double minWmc = Double.POSITIVE_INFINITY, maxWmc =
Double.NEGATIVE_INFINITY;
```

```
        for (var row : dataset.rows()) {
            empCbo.getData().add(new
XYChart.Data<>(row.cbo(), row.rfc()));
            empWmc.getData().add(new
XYChart.Data<>(row.wmc(), row.rfc()));

            minCbo = Math.min(minCbo, row.cbo());
            maxCbo = Math.max(maxCbo, row.cbo());
            minWmc = Math.min(minWmc, row.wmc());
            maxWmc = Math.max(maxWmc, row.wmc());
        }

        chartCbo.getData().add(empCbo);
        chartWmc.getData().add(empWmc);
        double cboInput = safeParseOr(8.625, "CBO");
        double wmcInput = safeParseOr(22.25, "WMC");
        double pConf = safeParseOr(0.95, "Pconf");

        plotCurvesOverRange(chartCbo, true, minCbo, maxCbo,
cboInput, wmcInput, pConf);
        plotCurvesOverRange(chartWmc, false, minWmc,
maxWmc, cboInput, wmcInput, pConf);
    }

    private void plotCurvesOverRange(LineChart<Number,
Number> chart, boolean varyCbo,
double minX, double maxX,
double cboFix, double wmcFix,
double pConf) {

        XYChart.Series<Number, Number> main = new
XYChart.Series<>();
        main.setName("RFC (оцінка)");

        XYChart.Series<Number, Number> confL = new
XYChart.Series<>();
        confL.setName("Довірчий (нижня)");

        XYChart.Series<Number, Number> confU = new
XYChart.Series<>();
        confU.setName("Довірчий (верхня)");

        XYChart.Series<Number, Number> predL = new
XYChart.Series<>();
        predL.setName("Прогноз (нижня)");
```

```
        XYChart.Series<Number, Number> predU = new
XYChart.Series<>();
        predU.setName("Прогноз (верхня)");

        int points = 60;
        double pad = (maxX - minX) * 0.05;
        double start = Math.max(1e-9, minX - pad);
        double end = maxX + pad;

        for (int i = 0; i < points; i++) {
            double x = start + (end - start) * i / (points
- 1.0);

            double cbo = varyCbo ? x : cboFix;
            double wmc = varyCbo ? wmcFix : x;

            var res = RfcModel.evaluate(dataset, cbo, wmc,
pConf);

            main.getData().add(new XYChart.Data<>(x,
res.rfc()));
            confL.getData().add(new XYChart.Data<>(x,
res.confLow()));
            confU.getData().add(new XYChart.Data<>(x,
res.confHigh()));
            predL.getData().add(new XYChart.Data<>(x,
res.predLow()));
            predU.getData().add(new XYChart.Data<>(x,
res.predHigh()));
        }

chart.getData().add(main);
chart.getData().add(confL);
chart.getData().add(confU);
chart.getData().add(predL);
chart.getData().add(predU);
    }

    private static TitledPane titled(String title,
javafx.scene.Node content) {
        TitledPane tp = new TitledPane(title, content);
        tp.setCollapsible(false);
        return tp;
    }
```

```
    private static LineChart<Number, Number>
createChart(String title, String xLabel, String yLabel) {
        NumberAxis x = new NumberAxis();
        NumberAxis y = new NumberAxis();
        x.setLabel(xLabel);
        y.setLabel(yLabel);

        LineChart<Number, Number> chart = new
LineChart<>(x, y);
        chart.setTitle(title);
        chart.setCreateSymbols(true); // важный момент:
чтобы были "кружочки" как в Scilab
        chart.setAnimated(false);
        chart.setLegendVisible(true);
        chart.setMinHeight(320);
        return chart;
    }

    private static double parseDouble(TextField tf, String
name) {
        try {
            String s = tf.getText().trim().replace(',',
'.');
            return Double.parseDouble(s);
        } catch (Exception e) {
            throw new IllegalArgumentException("Некоректне
значення для поля: " + name);
        }
    }

    private static void showError(String header, String
message) {
        Alert alert = new Alert(Alert.AlertType.ERROR);
        alert.setTitle("Помилка");
        alert.setHeaderText(header);
        alert.setContentText(message);
        alert.showAndWait();
    }

    private static void info(String message) {
        Alert a = new Alert(Alert.AlertType.INFORMATION);
        a.setTitle("OK");
        a.setHeaderText(null);
        a.setContentText(message);
        a.showAndWait();
    }
```

```
    private double safeParseOr(double def, String name) {
        return def;
    }
}
```

RfcModel.java:

```
package ua.rfcapp;

import org.apache.commons.math3.distribution.TDistribution;
import org.apache.commons.math3.linear.*;

import java.io.BufferedReader;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Path;
import java.util.ArrayList;
import java.util.List;

public final class RfcModel {

    private RfcModel() {}

    public record Row(double rfc, double cbo, double wmc)
{}

    public record Dataset(
            List<Row> rows,
            int n,
            double z1Mean,
            double z2Mean,
            RealMatrix szzInv,
            double b0,
            double b1,
            double b2,
            double sigmaLog,
            double r2
    ) {}

    public record Result(
            double rfc,
            double confLow,
            double confHigh,
            double predLow,
```

```
            double predHigh,
            double r2,
            double sigmaLog,
            double tValue,
            int df,
            double b0,
            double b1,
            double b2
    ) {}

    public static Dataset loadDataset(Path file) throws
IOException {
        List<Row> rows = new ArrayList<>();

        try (BufferedReader br =
Files.newBufferedReader(file, StandardCharsets.UTF_8)) {
            String line;
            int lineNo = 0;
            while ((line = br.readLine()) != null) {
                lineNo++;
                line = line.trim();
                if (line.isEmpty()) continue;
                line = line.replace(',', '.');
                String[] parts = line.split("\\s+");
                if (parts.length < 3) {
                    throw new IOException("Рядок " + lineNo
+ " має менше 3 чисел.");
                }

                double rfc = Double.parseDouble(parts[0]);
                double cbo = Double.parseDouble(parts[1]);
                double wmc = Double.parseDouble(parts[2]);

                if (rfc <= 0 || cbo <= 0 || wmc <= 0) {
                    throw new IOException("Рядок " + lineNo
+ ": RFC/CBO/WMC повинні бути > 0 (для log10).");
                }

                rows.add(new Row(rfc, cbo, wmc));
            }
        }

        int n = rows.size();
        if (n < 5) throw new IOException("Замало рядків у
файлі (потрібно хоча б 5).");
```

```
        // Средние z1A, z2A (z = log10)
        double z1Sum = 0, z2Sum = 0;
        for (Row r : rows) {
            z1Sum += log10(r.cbo());
            z2Sum += log10(r.wmc());
        }
        double z1Mean = z1Sum / n;
        double z2Mean = z2Sum / n;
        double sz1z1 = 0, sz2z2 = 0, sz1z2 = 0;
        for (Row r : rows) {
            double dz1 = log10(r.cbo()) - z1Mean;
            double dz2 = log10(r.wmc()) - z2Mean;
            sz1z1 += dz1 * dz1;
            sz2z2 += dz2 * dz2;
            sz1z2 += dz1 * dz2;
        }
        RealMatrix szz = MatrixUtils.createRealMatrix(new
double[][]{
                {sz1z1, sz1z2},
                {sz1z2, sz2z2}
        });

        RealMatrix szzInv = new
LUDecomposition(szz).getSolver().getInverse();
        y = log10(RFC), x1 = log10(CBO), x2 = log10(WMC)
        RealMatrix X = MatrixUtils.createRealMatrix(n, 3);
        RealVector Y = new ArrayRealVector(n);

        for (int i = 0; i < n; i++) {
            Row r = rows.get(i);
            X.setEntry(i, 0, 1.0);
            X.setEntry(i, 1, log10(r.cbo()));
            X.setEntry(i, 2, log10(r.wmc()));
            Y.setEntry(i, log10(r.rfc()));
        }

        beta = (X'X)^-1 X' y
        RealMatrix XtX = X.transpose().multiply(X);
        RealVector XtY = X.transpose().operate(Y);
        RealVector beta = new
LUDecomposition(XtX).getSolver().solve(XtY);

        double b0 = beta.getEntry(0);
        double b1 = beta.getEntry(1);
        double b2 = beta.getEntry(2);
        double sseLog = 0.0;
```

```
        double yMean = 0.0;
        for (Row r : rows) yMean += r.rfc();
        yMean /= n;

        double syA = 0.0;
        double syy = 0.0;

        for (Row r : rows) {
            double zPred = b0 + b1 * log10(r.cbo()) + b2 *
log10(r.wmc());
            double zObs = log10(r.rfc());
            double e = zPred - zObs;
            sseLog += e * e;

            syA += (r.rfc() - yMean) * (r.rfc() - yMean);

            double yPred = pow10(zPred);
            syy += (r.rfc() - yPred) * (r.rfc() - yPred);
        }

        int df = n - 3;
        double sigmaLog = sseLog / df;
        double r2 = 1.0 - (syy / syA);

        return new Dataset(rows, n, z1Mean, z2Mean, szzInv,
b0, b1, b2, sigmaLog, r2);
    }
    public static Result evaluate(Dataset ds, double cbo,
double wmc, double pConf) {
        if (cbo <= 0 || wmc <= 0) throw new
IllegalArgumentException("CBO і WMC повинні бути > 0.");
        if (pConf <= 0 || pConf >= 1) throw new
IllegalArgumentException("Довірча ймовірність повинна бути
між 0 та 1.");

        int n = ds.n();
        int df = n - 3;

        double alpha = 1.0 - pConf;
        double p = 1.0 - alpha / 2.0;
        double t = new
TDistribution(df).inverseCumulativeProbability(p);

        double z = ds.b0() + ds.b1() * log10(cbo) + ds.b2()
* log10(wmc);
```

```
        double dz1 = log10(cbo) - ds.z1Mean();
        double dz2 = log10(wmc) - ds.z2Mean();

        RealMatrix zi =
MatrixUtils.createColumnRealMatrix(new double[]{dz1, dz2});
        double leverage =
zi.transpose().multiply(ds.szzInv()).multiply(zi).getEntry(
0, 0);

        double deltaZ   = t * Math.sqrt(ds.sigmaLog() *
(1.0 / n + leverage));
        double deltaZpr = t * Math.sqrt(ds.sigmaLog() *
(1.0 + 1.0 / n + leverage));

        double zL   = z - deltaZ;
        double zU   = z + deltaZ;
        double zLpr = z - deltaZpr;
        double zUpr = z + deltaZpr;

        return new Result(
                pow10(z),
                pow10(zL),
                pow10(zU),
                pow10(zLpr),
                pow10(zUpr),
                ds.r2(),
                ds.sigmaLog(),
                t,
                df,
                ds.b0(),
                ds.b1(),
                ds.b2()
        );
    }

    public static double log10(double v) {
        return Math.log(v) / Math.log(10.0);
    }

    public static double pow10(double v) {
        return Math.pow(10.0, v);
    }
}
```

Pom.xml:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"

 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">

  <modelVersion>4.0.0</modelVersion>

  <groupId>ua.rfcapp</groupId>
  <artifactId>rfc-javafx</artifactId>
  <version>1.0.0</version>

  <properties>
    <maven.compiler.release>17</maven.compiler.release>
    <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
    <javafx.version>21.0.2</javafx.version>
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.openjfx</groupId>
      <artifactId>javafx-controls</artifactId>
      <version>${javafx.version}</version>
    </dependency>
    <dependency>
      <groupId>org.openjfx</groupId>
      <artifactId>javafx-graphics</artifactId>
      <version>${javafx.version}</version>
    </dependency>

    <dependency>
      <groupId>org.apache.commons</groupId>
      <artifactId>commons-math3</artifactId>
      <version>3.6.1</version>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.openjfx</groupId>
        <artifactId>javafx-maven-plugin</artifactId>
        <version>0.0.8</version>
```

```
        <configuration>
          <mainClass>ua.rfcapp.MainApp</mainClass>
        </configuration>
      </plugin>
    </plugins>
  </build>

</project>
```

ДОДАТОК В – Опис програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java

Загальні відомості

Програма для оцінювання метрики RFC у Java-застосунках з відкритим вихідним кодом реалізована як окремий програмний засіб із графічним інтерфейсом користувача, створеним мовою Java із використанням бібліотеки JavaFX.

Розроблений програмний засіб призначений для автоматизованого визначення якості програмних систем електронної комерції на Java на основі нелінійної регресійної моделі, що використовує метрики RFC, CBO та WMC.

Програма реалізована у вигляді Java-застосунку з використанням системи збирання Apache Maven. Структура проєкту включає програмний код, конфігураційні файли та необхідні ресурси, що забезпечують коректний запуск і стабільне функціонування застосунку.

Функціональне призначення

Програма призначена для оцінювання метрики RFC у Java-застосунках з відкритим вихідним кодом шляхом використання нелінійної регресійної моделі, побудованої на основі емпіричних значень метрик CBO та WMC.

Основні функції програмного засобу включають:

- імпорт емпіричних даних метрик RFC, CBO та WMC із текстового файлу;
- введення користувачем значень метрик CBO та WMC для програмної системи, що аналізується;
- встановлення значення довірчої ймовірності;
- обчислення прогнозованого значення метрики RFC;
- формування довірчих та прогнозних інтервалів для метрики RFC;

- визначення статистичних характеристик моделі, зокрема коефіцієнта детермінації R^2 та середньої відносної похибки MRE (за умови введення фактичного значення RFC);
- візуалізацію результатів оцінювання у вигляді графіків залежності RFC від метрик CBO та WMC;
- відображення всіх результатів оцінювання у графічному інтерфейсі користувача.
-

Технічні засоби, що використовуються

Програма повинна експлуатуватися на обчислювальному обладнанні з параметрами не нижчими за такі:

- процесор рівня Intel Core i3 або еквівалентний;
- оперативна пам’ять обсягом не менше 2 ГБ;
- вільний простір на жорсткому диску не менше 600 МБ.
- Виклик та завантаження

Для запуску та роботи з програмою необхідна операційна система Windows 10 або Windows 11 (64-бітна), встановлене середовище виконання Java Development Kit (JDK) версії 17 або вище, а також бібліотека JavaFX.

Запуск програми здійснюється шляхом виконання зібраного Java-застосунку або через середовище розробки, яке підтримує Maven-проєкти. Після запуску на екран виводиться головне вікно програми з елементами керування для введення даних, запуску оцінювання та перегляду результатів.

ДОДАТОК Г – Інструкція користувача програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java

Загальний опис роботи з програмою

Після запуску JavaFX-застосунку на екрані відображається головне вікно програми для оцінювання метрики RFC застосунків з відкритим кодом на Java, яке відповідає початковому стану програмного засобу (рисунок Г.1).

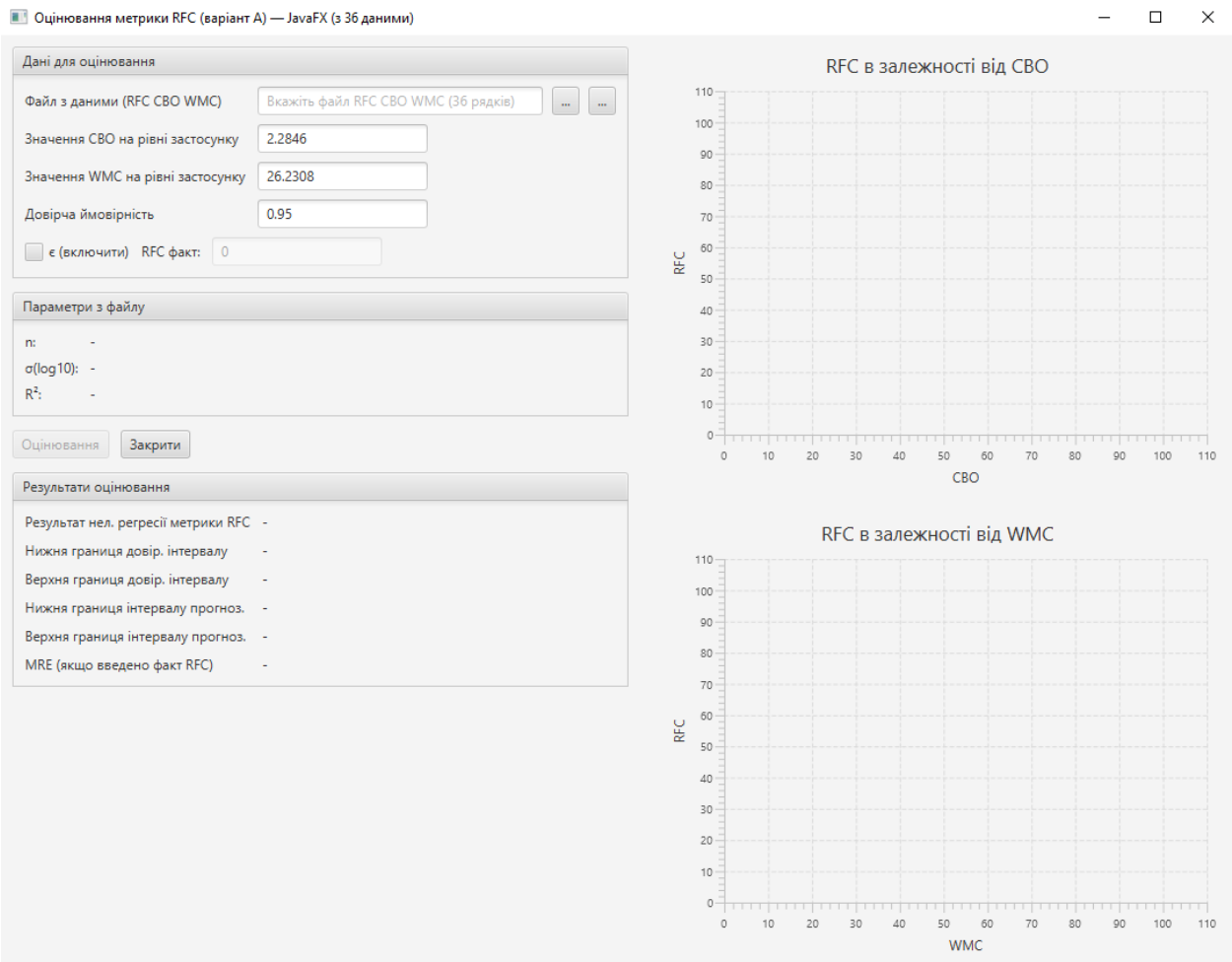

Рисунок Г.1 – Головне вікно програми для оцінювання метрики RFC

У головному вікні програми користувачеві необхідно вибрати файл з початковими емпіричними даними, що містить значення метрик RFC, CBO та WMC застосунків з відкритим кодом на Java. Вибір файлу здійснюється за допомогою відповідного елемента керування «Файл з даними».

Після вибору файлу програма автоматично зчитує емпіричні дані та відображає параметри вибірки, зокрема кількість рядків, оцінку стандартного відхилення залишків у логарифмічному просторі та значення коефіцієнта детермінації R^2.

Далі у відповідних полях введення користувач повинен задати значення метрик CBO та WMC застосунку, для якого виконується оцінювання метрики RFC, а також встановити значення довірчої ймовірності.

Крім того, у головному вікні програми користувач може за допомогою відповідного елемента керування вказати наявність фактичного значення метрики RFC. У разі вибору цього режиму необхідно ввести фактичне значення RFC застосунку для подальшого порівняння з отриманою оцінкою та обчислення середньої відносної похибки MRE.

Після введення всіх необхідних даних користувачеві потрібно натиснути на командну кнопку «Оцінювання». У результаті виконання обчислень у програмі відображаються:

- прогнозоване значення метрики RFC;
- нижня та верхня межі довірчого інтервалу;
- нижня та верхня межі інтервалу прогнозування;
- значення коефіцієнта детермінації R^2;
- значення середньої відносної похибки MRE (у разі наявності фактичного RFC).

Окрім числових результатів, програма автоматично будує графіки залежності метрики RFC від метрик CBO та WMC, які відображають емпіричні дані, регресійну оцінку, а також довірчі та прогнозні інтервали.

Після завершення роботи з програмою користувач може натиснути командну кнопку «Закрити», що призводить до закриття головного вікна програми та завершення виконання JavaFX-застосунку.

ДОДАТОК Д – Програма та методика випробувань програмної системи для автоматизації прийняття рішень щодо якості ПС e-commerce на Java

Об’єкт випробувань

Об’єктом випробувань є програмний засіб для оцінювання метрики RFC програмних систем з відкритим кодом на Java, реалізований у вигляді JavaFX-застосунку.

Мета випробувань

Метою випробувань є перевірка відповідності програмного засобу вимогам технічного завдання, коректності реалізації математичної моделі, а також правильності обчислення показників оцінювання якості програмних систем.

Склад програмної документації

До складу програмної документації входять:

- технічне завдання;
- опис програми;
- інструкція користувача;
- текст програми;
- програма та методика випробувань.
-

Технічні вимоги

Програмний засіб повинен коректно функціонувати на обчислювальному обладнанні з характеристиками, наведеними в описі програми, та забезпечувати стабільну роботу графічного інтерфейсу користувача і математичного модуля обчислень.

Методи випробувань

Випробування програмного засобу проводяться шляхом тестування за методологією «чорної скриньки» (рис. Д.1 – Д.4).

У процесі випробувань виконується:

- запуск програмного засобу;
- перевірка коректності роботи всіх елементів графічного інтерфейсу;
- завантаження емпіричних даних із файлу;
- виконання процедури оцінювання метрики RFC;
- аналіз отриманих числових результатів та побудованих графіків.

У разі відповідності отриманих результатів вимогам технічного завдання та очікуваним значенням, програмний засіб вважається таким, що пройшов випробування та придатний до впровадження в експлуатацію. У протилежному випадку програмний засіб передається на доопрацювання розробнику.

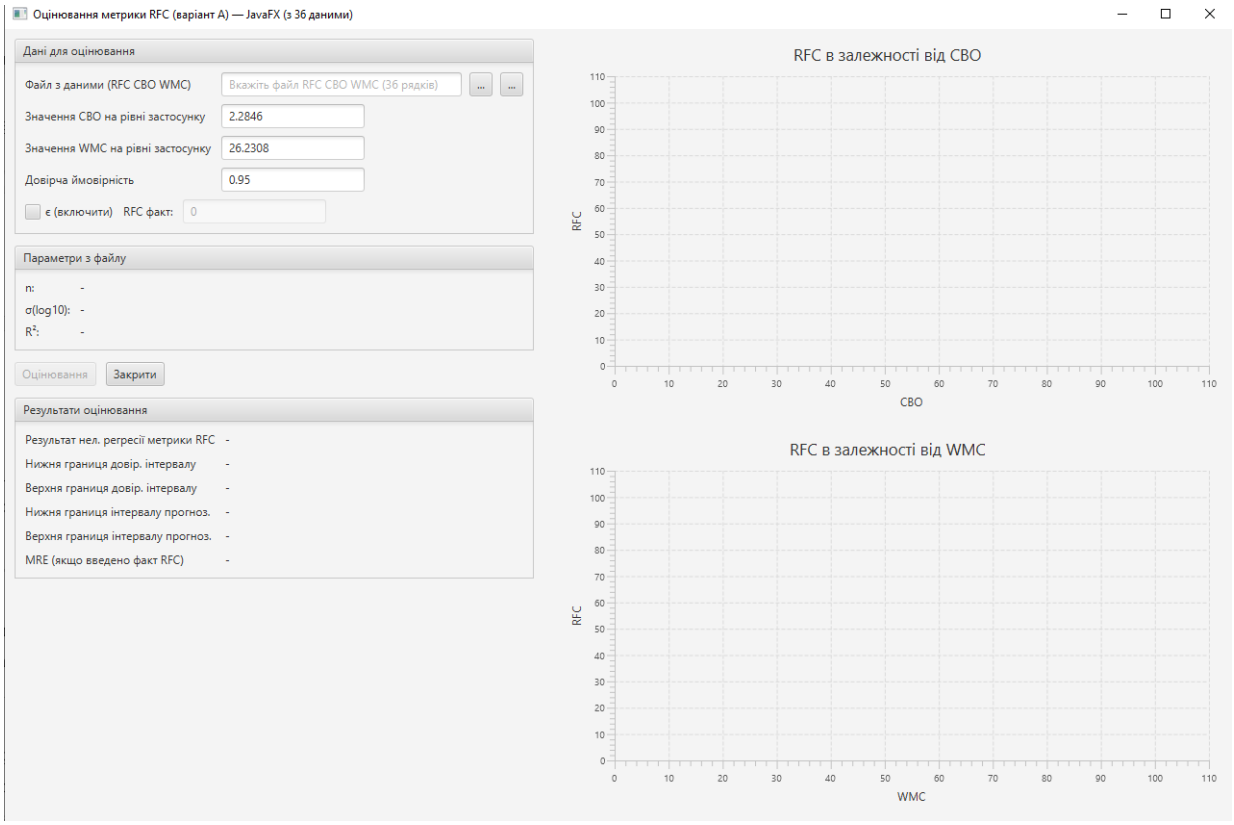

Рисунок Д.1 – Графічне вікно головного модуля програми

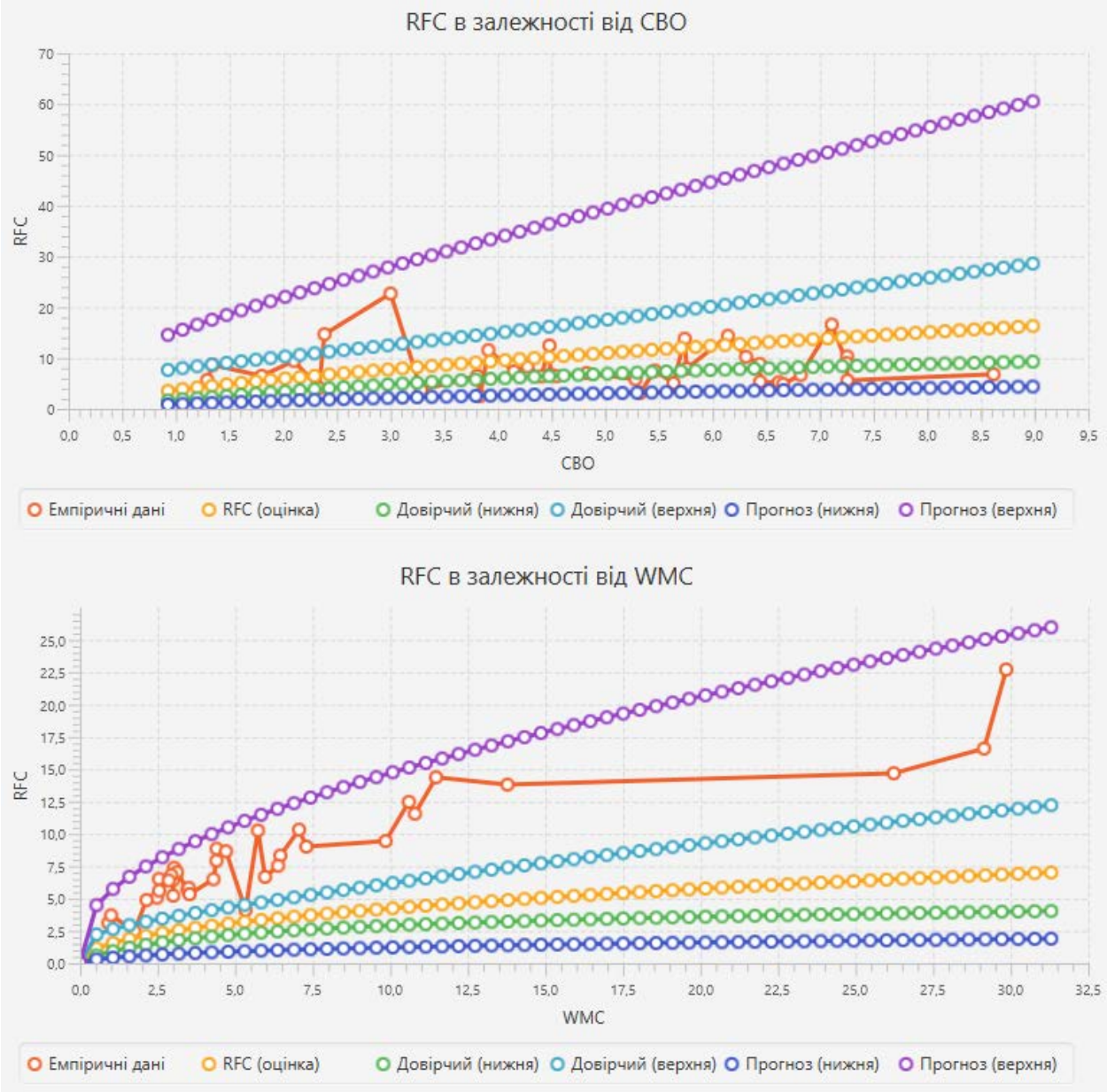

Рисунок Д.2 – Візуалізація тестових емпіричних даних

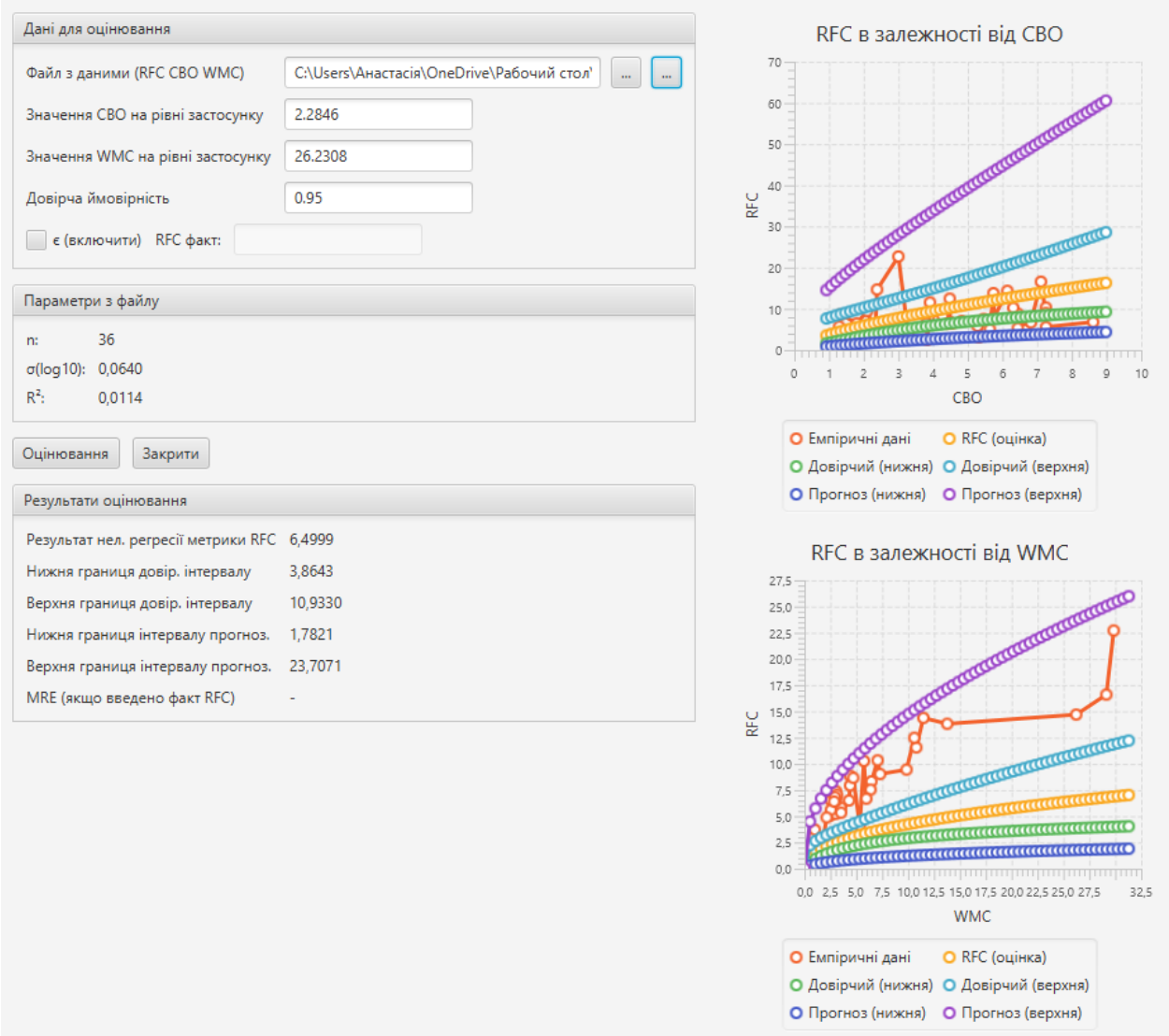

Рисунок Д.3 – Результати роботи програмного засобу після виконання команди «Оцінювання» без введення фактичного значення метрики RFC

Рисунок Д.4 – Результати роботи програмного засобу після виконання команди «Оцінювання» з увімкненим режимом введення фактичного значення метрики RFC

При випробуванні програми необхідно ввести наступні тестові дані з текстового файлу, що містить емпіричні значення метрик RFC, CBO та WMC для 36 програмних проєктів з відкритим кодом, розроблених мовою Java:

5.7022 1.2921 3.4607

14.6923 2.3846 26.2308

6.814 4.4884 3.093

6.5 1.8 4.3

14.381 6.1429 11.4762

3.1176 5.3235 0.9118

11.5588 3.9118 10.7941

10.3284 7.2537 7.0597

6.5181 3.8072 2.5422

5.8296 5.2815 3.5037

5.0682 5.6364 2.4773

6.4318 4.5455 3

5.2308 6.6154 3

7.3871 4.1505 3.0323

8.8571 6.44 4.4

9.0227 7.25 7.2955

2.5 3.8333 1.75

8.3246 4.2761 6.4552

5.6098 7.2561 2.5488

7.931 5.7586 4.3966

10.2717 6.3121 5.7341

7.052 4.8266 3.1214

12.48 4.48 10.6

13.8148 5.7407 13.7778

4.1111 2.3333 5.3333

7.5385 5.4615 6.3846

4.8857 6.6571 2.1429

6.8235 8.6176 2.9412

16.6014 7.1104 29.1408

3.7143 3.2857 1

6.4051 4.4051 2.8608

9.45 2.1 9.85

22.7143 3 29.8571

8.6591 1.3409 4.7045

6.6786 6.8214 5.9643

5.3429 6.4429 3.5286