Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи)

магістра
(освітній ступінь (освітньо-кваліфікаційний рівень))

на тему: **Методи та засоби оцінювання надійності комп'ютерних систем при збоях програмного забезпечення**

Виконав: студент (ка) 6 курсу, групи СІм-61
спеціальності 123
«Комп'ютерна інженерія»
(шифр і назва спеціальності (напряму підготовки))
(підпис) Юськів Я.І.
(прізвище та ініціали)
Керівник (підпис) Тиш Є.В.
(прізвище та ініціали)
Нормоконтроль (підпис) Луцик Н.С.
(прізвище та ініціали)
Рецензент (підпис) Гаврилюк Н.Б.
(прізвище та ініціали)

м. Тернопіль – 2019

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)

Факультет комп'ютерно-інформаційних систем і програмної інженерії
Кафедра комп'ютерних систем та мереж
Освітній ступінь магістр
Напрям підготовки
(шифр і назва)
Спеціальність 123 Комп'ютерна інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри Осухівська Г.М.
«30» 04 2019 р.

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Юзківу Ярославу Івановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи та засоби оцінювання надійності комп'ютерних систем при збоях програмного забезпечення

Керівник роботи к.т.н., доц. Тиш Євгенія Володимирівна
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом по університету від «21» 03 2019 року №4/7-254
2. Термін подання студентом роботи 24 грудня 2019
3. Вихідні дані до проекту роботи стан розвитку програмного забезпечення, критерії надійності комп'ютерних систем

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Вступ 1. Аналіз сучасного стану розвитку методів і засобів забезпечення надійності комп'ютерних систем. 2. Побудова моделі для розрахунку впливу помилок програмного забезпечення на надійність комп'ютерних систем. 4. Розр[illegible]. Аналіз впливу збоїв програмного забезпечення на надійність комп'ютерних систем. 5. Обґрунтування економічної ефективності. 6. Охорона праці та безпека в надзвичайних ситуаціях. 7. Екологія

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів) 1. Методи та засоби оцінювання надійності комп'ютерних систем при збоях ПЗ 2. Об'єкт, предмет, мета [illegible] 3. Структура комп'ютерних систем і [illegible] 4. [illegible] програмного забезпечення 5. [illegible] 6. Модель оцінювання надійності комп'ютерних систем при збоях ПЗ 7. [illegible] засобу оцінювання надійності комп'ютерних систем на ER-діаграмі бази даних 8. Результати оцінювання збоїв ПЗ

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата: завдання видав	Підпис, дата: завдання прийняв
Економіка	Косова О. М., доц.		
Обґрунтування економічної ефективності	Кузич Н. Б.		
Безпека в НС	Струтинська В.С., ст. викл. каф.		
Охорона праці	Будясов П. И.	21.11	19.11

7. Дата видачі завдання 30.09.2019

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз сучасного стану методів і засобів забезпечення надійності калібрувальних систем	1.10.19	виконано
2.	Побудова моделі та розробка методу аналізу впливу наслідків програмного забезпечення на надійність калібрувальних систем	21.10.19	виконано
3.	Засоби підтримки процесу визначення впливу засобів програмного забезпечення на надійність калібрувальних систем	4.11.19	виконано
4	Обґрунтування економічної ефективності	11.11.19	виконано
5.	Охорона праці та безпека в надзвичайних ситуаціях	12.11.19	виконано
6.	Економіка	25.11.19	виконано
	Попередній захист дипломної роботи	28.11.19	виконано
	Захист дипломної роботи	26.12.19	виконано

Студент ______ (підпис) Юлків Я. Ї. (прізвище та ініціали)

Керівник роботи ______ (підпис) Бліщ Б. В. (прізвище та ініціали)

АНОТАЦІЯ

Тема дипломної роботи: “ Методи та засоби оцінювання надійності комп’ютерних систем при збоях програмного забезпечення ”. // Дипломна робота // Юськів Ярослав Іванович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп’ютерно-інформаційних систем та програмної інженерії, група СІм-61 // Тернопіль, 2019 // с. – 128 , рис. – 32 , табл. – 11, аркушів А1 – 10 , додат. –2 , бібліогр. – 28 .

Ключові слова: ЗБІЙ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, КОМП’ЮТЕРНА СИСТЕМА, НАДІЙНІСТЬ, ВПЛИВ.

У першому розділі дипломної роботи розглянуто принципи організації та функціонування сучасних комп’ютерних систем і виявлено, що основними їхніми компонентами є апаратне, програмне забезпечення та канали зв’язку, причому роль програмного забезпечення є визначальною у забезпеченні функціональності та зручності використання системи. Проведено аналіз типів помилок, дефектів та відмов, які притаманні програмному забезпеченню комп’ютерних систем, розглянуто моделі надійності комп’ютерних систем та обгрунтовано необхідність дослідження методів і засобів виявлення дефектів у програмному забезпеченні та оцінюванні його впливу на надійність функціонування та експлуатації комп’ютерних систем. Обґрунтовано необхідність застосування моделей раннього і пізнього прогнозування помилок програмного забезпечення при проектуванні комп’ютерних систем для того, щоб побудувати комплексний підхід до визначення впливу дефектів програмного забезпечення на надійність експлуатації комп’ютерних систем з врахуванням історії виникнення помилок у ПЗ.

У другому розділі дипломної роботи проведено формалізацію показників надійності програмних складових комп’ютерних систем, побудовано модель визначення впливу збоїв програмного забезпечення на надійність комп’ютерних систем, обгрунтовано моделі представлення дефектів програмного забезпечення та метод моніторингу за їх розвитком, обгрунтовано моделі раннього і пізнього

прогнозування дефектів комп’ютерних систем. Запропонована модель визначення впливу дефектів програмного забезпечення на надійність комп’ютерної системи забезпечує можливість встановлення шляху поширення дефекту програмного забезпечення на інші компоненти комп’ютерної системи і дає змогу прогнозувати ймовірність виникнення негативного впливу на надійність комп’ютерної системи в цілому. Обгрунтовані модель і метод моніторингу дефектів програмного забезпечення комп’ютерних систем на основі модифікованої моделі Hismo дає змогу аналізувати розвиток і вплив дефекту програмного забезпечення на надійність комп’ютерної системи у часі та враховувати версії програмного забезпечення.

У третьому розділі за допомогою мови моделювання UML та use case діаграм визначено функціональні вимоги до програмного засобу підтримки процесу визначення впливу дефектів програмного забезпечення на надійність комп’ютерних систем, спроектовано базу даних для зберігання та маніпулювання даними при визначенні впливу дефектів програмного забезпечення на надійність комп’ютерної системи та реалізовано її у середовищі MS SQL Server. Спроектовано архітектуру та розроблено інтерфейси користувачів програмного засобу маніпулювання критеріями надійності, моделями прогнозування дефектів програмного забезпечення, формування імовірних шляхів поширення дефектів на компоненти комп’ютерної системи, а також проведено експериментальні дослідження щодо застосування запропонованих та обґрунтованих у роботі моделей, методів і програмного засобу, що дало змогу підтвердити доцільність їх використання.

У четвертому розділі проведено розрахунки для визначення доцільності проведення науково-дослідної роботи щодо методів і засобів оцінювання впливу збоїв програмного забезпечення на надійність комп’ютерних систем і встановлено, що собівартість запропонованих рішень становить 50342,69 грн., а термін їхньої окупності – 1,68 року, що дає змогу обґрунтувати економічну ефективність одержаних результатів.

У п’ятому розділі проведено аналіз вимог з охорони праці і техніки безпеки при роботі з програмним засобом підтримки процесу оцінювання надійності комп’ютерних систем при збоях програмного забезпечення, визначено шляхи мінімізації негативного впливу шкідливих факторів на здоров’я користувачів ПК, а також застосування режимів захисту і хімічного контролю при викиді НХР у випадку аварії на хімічно небезпечному об’єкті та джерел виникнення шуму і вібрацій.

У шостому розділі проаналізовано методи і засоби формування бази статистичних даних в екології та використання в Україні альтернативних джерел енергії.

ABSTRACT

The theme of the thesis: " Methods and tools of computer systems reliability assessment at software failures " //Master work // Yuskiv Yaroslav Ivanovych/ Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and software engineering, group CIm-61 // Ternopil, 2019 // p. –128, fig. – 32, table – 11, sheets A1 – 10, Add – 2, Ref. – 28.

KEYWORDS: FAILURE, SOFTWARE, RELIABILITY, COMPUTER SYSTEM, IMPACT.

The first chapter of the thesis examines the principles of organization and operation of modern computer systems and found that their main components are hardware, software and communication channels, and the role of software is crucial in ensuring the functionality and ease of use of the system. The types of errors, defects and failures inherent in the software of computer systems are analyzed, the models of reliability of computer systems are considered and the necessity of research of methods and means of detection of defects in the software and evaluation of its influence on the reliability of the functioning and operation of computer systems is substantiated . The necessity to apply models of early and late prediction of software errors in the design of computer systems in order to build a comprehensive approach to determine the impact of software defects on the reliability of the operation of computer systems, taking into account the history of errors in the software.

In the second chapter of the diploma paper formalization of reliability indicators of software components of computer systems is carried out, the model of determining the impact of software failures on the reliability of computer systems, the models of presentation of software defects and the method for monitoring their development, models of early and late prediction of defects are substantiated computer systems. The proposed model for determining the impact of software defects on the reliability of the computer system provides an opportunity to determine the path of propagation of the

software defect to other components of the computer system and allows to predict the likelihood of a negative impact on the reliability of the computer system as a whole. A well-grounded model and method for monitoring computer system software defects based on a modified Hismo model allows us to analyze the development and impact of a software defect on the reliability of a computer system over time and to consider software versions.

The third chapter uses the UML modeling language and use case diagrams to determine the functional requirements for the software to support the impact of software defects on the reliability of computer systems, to design a database for storing and manipulating data to determine the effects of software defects on the reliability of computers. and implemented in MS SQL Server. The architecture is designed and user interfaces of the software are manipulated by the criteria of reliability, models of forecasting of defects of the software, formation of probable ways of spreading of defects on the components of the computer system, and also experimental researches on application of the offered models and methods, methods and programs are carried out. allowed us to confirm the appropriateness of their use.

In the fourth chapter, calculations were made to determine the feasibility of conducting research on methods and means of assessing the impact of software failures on the reliability of computer systems, and found that the cost of the proposed solutions is 50342,69 UAH, and their payback period - 1,68 year, which allows to substantiate the economic efficiency of the results obtained.

The fifth chapter analyzes the requirements for occupational safety and health when working with software to support the process of assessing the reliability of computer systems in case of software crashes, identifying ways to minimize the negative impact of harmful factors on the health of PC users, as well as the application of modes protection and chemical control during the release of NHR in the event of an accident at a chemical hazardous object and sources of noise and vibration.

The sixth chapter analyzes the methods and means of forming a database of statistics in ecology and the use of alternative energy sources in Ukraine.

ЗМІСТ

ПЕРЕЛІК ОСНОВНИХ УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ І СКОРОЧЕНЬ

ЖЦ	Життєвий цикл
ПЗ	Програмний засіб (або програмне забезпечення системи)
ПП	Програмний продукт
КС	Комп’ютерна система
RLM	Rome Reliability Model
UML	Unified Modeling Language

ВСТУП

Актуальність теми. Сучасні комп’ютерні системи представляють собою сукупність програмно-апаратних засобів, які взаємодіють через визначені канали зв’язку і забезпечують необхідний рівень функціональності, надійності, продуктивності, зручності використання та інших характеристик. При цьому складність, розподіленість та багатозадачність комп’ютерних систем обумовлює посилення вимог до надійності їхнього функціонування у заданому контексті використання.

Оскільки, до складу комп’ютерних систем входить апаратне забезпечення, вбудоване та високорівневе програмне забезпечення, то важливими задачами в галузі забезпечення та моніторингу надійності комп’ютерних систем є дослідження впливу помилок, внесених у компоненти системи.

Наслідком невиявлених, в процесі розробки комп’ютерних систем, помилок є розвиток і поширення дефектів апаратного і програмного забезпечення на етапах їх експлуатації і супроводу. В свою чергу, це може призводити до відмов або збоїв системи, усунути які є доволі затратно як за часовими, так і фінансовими ресурсами.

Слід відмітити, що сучасний рівень розвитку методів і засобів тестування як апаратного, так і програмного забезпечення, дає змогу забезпечити виявлення дефектів на етапах життєвого циклу, однак внесені логічні помилки у програмне забезпечення можуть проявляти себе на завершальних етапах проектування або під час супроводу комп’ютерних систем. Тому, актуальною задачею, в контексті забезпечення надійності комп’ютерних систем, є розробка та впровадження методів виявлення, моніторингу та визначення впливу помилок програмного забезпечення на надійність комп’ютерної системи.

Дослідженню надійності комп’ютерних систем присвячено ряд наукових публікацій українських та закордонних вчених. Серед вітчизняних науковців необхідно відмітити вагомий внесок таких вчених як: Локазюк В.М., Лавріщева К.М., Коваль Г.І., Коротун Т.М., Волочій Б. Ю, Озірковський Л. Д.,

Сидоров М.О. та ін, серед закордонних – Lakey P.B., Musa J.D., Hecht H., Ohlsson N., Fenton N.E. та ін.

Враховуючи суттєвий внесок науковців у розвиток методів і засобів забезпечення та оцінювання надійності комп'ютерних систем, все ж не до кінця дослідженим залишається ряд питань щодо впливу і розвитку дефектів програмного забезпечення у часі, які можуть призвести до збоїв, на надійність комп'ютерних систем. Тому, задача дослідження впливу збоїв програмного забезпечення на надійність комп'ютерних систем є на сьогодні доволі актуальною задачею.

Мета роботи полягає у дослідженні моделей, методів і засобів виявлення, моніторингу та визначення впливу дефектів програмного забезпечення, які призводять до збоїв, на надійність комп'ютерних систем.

Об'єктом дослідження є процеси виявлення та аналізу впливу помилок програмного забезпечення на надійність комп'ютерних систем.

Предметом дослідження є моделі, методи і засоби виявлення та аналізу дефектів програмного забезпечення, моделі та методи представлення надійності комп'ютерних систем.

Задачі, які необхідно вирішити у магістерській роботі полягають у наступному:

– дослідження наукових публікацій і практик забезпечення надійності комп'ютерних систем;

– дослідження сучасних методів і засобів виявлення та аналізу дефектів програмного забезпечення комп'ютерних систем;

– побудова та обґрунтування моделей представлення дефектів програмного забезпечення;

– побудова та представлення моделей надійності комп'ютерних систем;

– розробка та обґрунтування методу аналізу впливу дефектів програмного забезпечення на надійність комп'ютерних систем;

– розроблення архітектури засобу аналізу дефектів програмного забезпечення комп'ютерних систем.

Наукова новизна одержаних результатів при виконанні дипломної роботи полягає в наступному:

– уперше побудовано та обґрунтовано модель впливу дефектів програмного забезпечення на надійність комп'ютерної системи, що забезпечує можливість встановлення шляху поширення дефекту програмного забезпечення на інші компоненти комп'ютерної системи і дає змогу прогнозувати ймовірність виникнення негативного впливу на надійність комп'ютерної системи.

– набули подальшого розвитку модель і метод моніторингу дефектів програмного забезпечення комп'ютерних систем на основі модифікованої моделі Hismo, що дає змогу аналізувати розвиток і вплив дефекту програмного забезпечення на надійність комп'ютерної системи у часі та з врахуванням версій програмного забезпечення.

Методи дослідження. При виконанні дипломної роботи магістра використовувались наступні методи: аналіз та синтез – під час опису дефектів програмного забезпечення комп'ютерних систем, їх аналізу та класифікації; формалізація і моделювання – під час побудови моделей дефектів програмного забезпечення та надійності комп'ютерних систем; об'єктно-орієнтовані аналіз, проектування та програмування – під час розроблення засобу аналізу дефектів програмного забезпечення; експеримент – під час апробації запропонованих методу та засобу.

Практична цінність результатів дослідження. Практична цінність роботи полягає у створенні архітектури та реалізації програмного комплексу підтримки методу визначення впливу дефектів програмного забезпечення на надійність комп'ютерних систем.

Публікації. Результати дослідження апробовано на VIII міжнародній науково - технічній конференції молодих учених і студентів «Актуальні задачі сучасних технологій» (27-28 листопада 2019 р.) Тернопільського національного технічного університету імені Івана Пулюя та на VII науково-технічній конференції Тернопільського національного технічного університету імені Івана

Пулюя «Інформаційні моделі, системи та технології» (11-12 грудня 2019 року) у вигляді тез конференцій.

1. Тиш Є.В., Юськів Я.І. Модель виявлення впливу дефектів програмного забезпечення на надійність комп'ютерних систем. Матеріали VII міжнародній науково - технічній конференції молодих учених і студентів «Актуальні задачі сучасних технологій» (27-28 листопада 2019 р.) Тернопільського національного технічного університету імені Івана Пулюя. Тернопіль: ТНТУ. 2019. с. 103-104.
2. Юськів Я., Тиш Є. База даних підтримки процесу оцінювання впливу дефектів програмного забезпечення на надійність комп'ютерних систем. Матеріали VII науково-технічної конференції Тернопільського національного технічного університету імені Івана Пулюя «Інформаційні моделі, системи та технології» (11-12 грудня 2019 року). Тернопіль: ТНТУ. 2019. С. 146.

Структура роботи. Робота складається з розрахунково-пояснювальної записки та графічної частини. Розрахунково-пояснювальна записка складається із вступу, 6 розділів, висновків, переліку посилань та додатків. Обсяг роботи: розрахунково-пояснювальна записка – 128 арк. формату А4, графічна частина – 10 аркушів формату А1.

РОЗДІЛ 1
АНАЛІЗ СУЧАСНОГО СТАНУ РОЗВИТКУ МЕТОДІВ І ЗАСОБІВ ЗАБЕЗПЕЧЕННЯ НАДІЙНОСТІ КОМП'ЮТЕРНИХ СИСТЕМ

1.1. Аналіз принципів організації та функціонування комп'ютерних систем

Комп'ютерна система – інформаційно-технічний комплекс, що здійснює опрацювання, збереження та ввід-вивід інформації [1]. У загальному випадку до складу комп'ютерної системи входять комп'ютери користувачів, сервери, периферійні пристрої, керування якими виконується програмним забезпеченням [1]. Каналами передачі даних між компонентами комп'ютерних систем виступають локальні або глобальні системи передачі даних. Структуру комп'ютерної системи, з точки зору функціонування, можна зобразити, як показано на рис. 1.1.

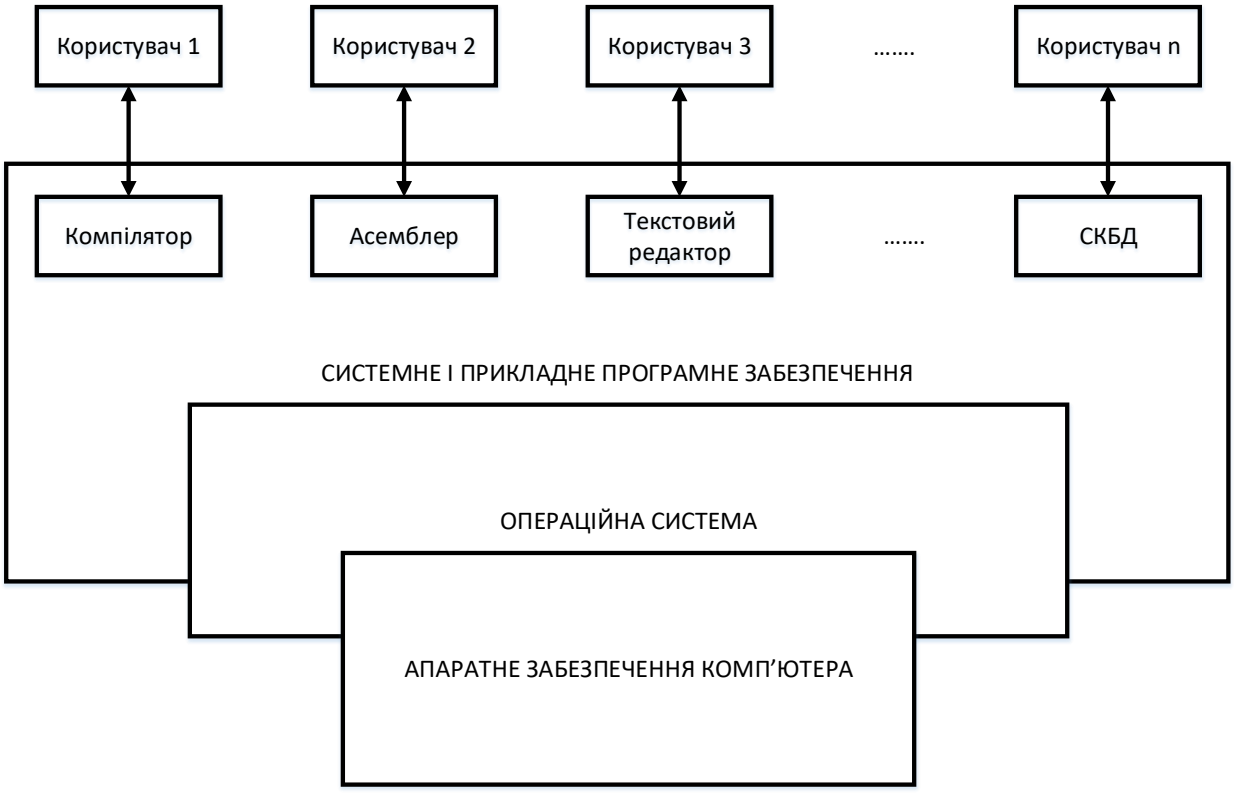

Рис. 1.1. Принцип функціонування комп'ютерної системи

Для опису комп’ютерних систем використовуються технічні, організаційні, документальні, функціональні, алгоритмічні, програмні та інформаційні структури [1]. Задачі, що розв’язуються в комп'ютерних інформаційних системах, мають ряд характерних особливостей, що впливають на технологію автоматизованого опрацювання інформації.

Користувачам комп’ютера доступні верхні рівні програмного забезпечення – системні та прикладні програми (наприклад, компілятори, текстові редактори, системи управління базами даних). Ці програми взаємодіють з операційною системою, яка, в свою чергу, керує роботою комп'ютера.

Один з видів класифікації комп’ютерних систем, що враховує комплексні характеристики комп’ютерів є:

– розмір;
– вік;
– мета;
– опрацювання;
– кількість процесорів.

За розмір комп’ютери поділяють на: суперкомп’ютери, мейнфрейми (main frame computers), міні комп’ютери, мікро комп’ютери, мобільні комп’ютери і вбудовані системи.

Суперкомп’ютери – виконують багато математичних розрахунків і генерують дуже велику кількість інформації. Вони також мають спеціалізовані програми, які потребують великих математичних розрахунків. Прикладами застосування суперкомп’ютерів є прогноз погоди, військові розрахунки, перевірка наслідків ядерної зброї. У Голлівуді суперкомп'ютери використовують для створення анімації, а дослідники-біологи застосовують їх для аналізу біологічних сполук, таких як кров. Деякі приклади включають Jaguar, Tianhe, Roadrunner та Sunway TaihuLight.

Мейнфрейми – це комп'ютери, дуже великі і дуже дорогі. Їх швидкість обробки даних надзвичайно висока. Це по суті міні суперкомп’ютери. Вони більш потужні, ніж суперкомп'ютери, тому що здатні одночасно виконувати

багато програм. Вони використовуються в онлайнових багатопотокових трансляціях, виконують операції інтернет-банкінгу. Деякі приклади включають Hitachi Z800 та System Z9.

Міні-комп’ютери – це комп’ютери, що здатні одночасно забезпечувати роботу близько 200 користувачів. Вони в основному використовуються для контролю та виробництва, моніторингу лабораторного обладнання, а також використовуються як засоби управління комутаторами. Деякі приклади включають Micro VAX II і Texas Instrument TI-990.

Мікро-комп’ютери – це комп’ютери, які за розміром трохи менші, ніж суперкомп'ютери та мейнфрейми. До них належать настільні та портативні комп'ютери.

Мобільні комп’ютери – включають мобільні телефони, ноутбуки та калькулятори.

Вбудовані системи – це електронні комп’ютери, призначені для виконання завдань. Вони представляють собою комбінацію програмного забезпечення, зовнішнього апаратного забезпечення та мікропроцесорного чіпа. Деякими прикладами вбудованих систем є принтери, цифрові камери, банкомати, термостати та цифрові годинники.

За цілями використання комп’ютери поділяються на дві групи. Комп'ютери загального призначення або багатоцільові вирішують широке коло завдань. На них можна встановити багато програм, які будуть вирішувати задачі з не високою швидкодією та ефективністю. Вони можуть бути використані для управління ресурсами підприємства, друку, підтримки бізнес-процесів продажу і т.п. До них належать ноутбуки та персональні комп'ютери.

Комп’ютери спеціального призначення – це комп'ютери, що вирішують строго визначені задачі. У них є набір вбудованих інструкцій, які вирішують нескладні задачі. До них відносяться калькулятори, машини для підрахунку грошей та ін.

Одноразові комп’ютери мають мінімальні розміри та невеликий набір команд, використовуються у радіочастотній ідентифікації (IMC RFID -мітки) для ідентифікації різних об'єктів та у побутових пристроях (музичні іграшки, листівки ті інші).

Вбудовані комп’ютери (мікроконтролери) не мають власного корпусу, вони монтуються в інші пристрої, мають певний набір команд, що запропонований заводом-розробником цих пристроїв. Використовуються в побутовій техніці, вимірювальних приладах, системах автоматичного керування і ін.

Ігрові автомати мають невеликі розміри, та своє ПЗ, можлива наявність монітора та жорсткого диску, не можливе переобладнання.

Персональні комп'ютери (ПК) настільні (стаціонарні) та переносні (портативні), мають складну ОС, широкий спектр програмного забезпечення та допускають переобладнання та модернізацію ПК.

Сервери – це комп’ютери для локальних мереж та доступу і роботи в мережі Інтернет. Мають декілька процесорів, великі об’єми ОЗП та жорсткого диску. Вони працюють у мережах із високою швидкістю.

Провівши аналіз особливостей застосування та структури комп’ютерних систем, а також враховуючи стрімкий розвиток інформаційних технологій у галузі комп’ютерної інженерії та інженерії програмного забезпечення, визначальними властивостями комп’ютерних систем є їх застосування і призначення. Архітектурні особливості комп’ютерних систем дають змогу реалізувати їх, використовуючи різні підходи і технології. Тому на ринку ІТ зараз доволі багато однотипних комп’ютерних систем, які вирішують подібні задачі із заданим рівнем надійності, однак обрати оптимальну систему з врахуванням вимог замовника і користувачів доволі складно.

Тому важливими задачами у контексті забезпечення надійності комп’ютерних систем є побудова моделі, яка б забезпечувала можливість встановлення впливу дефектів у компонентах програмного забезпечення на надійність комп’ютерних систем.

1.2. Аналіз та класифікація помилок, дефектів і відмов програмного забезпечення

Класифікацію помилок програмного забезпечення визначено та описано у стандарті ANSI/IEEE–729–83. В загальному випадку, у стандарті визначено такі три типи помилок [3]:

Помилка – ситуація, коли при виконанні програми одержують неправильні або не коректні результати у зв'язку з недоліками реалізації алгоритмів, пов'язаних з помилками використання операторів у процесі проектування та реалізації програми.

Дефект – тип помилки, що отриманий внаслідок неякісної реалізації програмних модулів зі сторони розробника. Дефекти можуть бути спричинені неточністю специфікації вимог до ПЗ, проектною та супровідною документацією, безпосередньо кодом програмних модулів. Дефекти виявляють як у процесі тестування, так і експлуатації програмного забезпечення комп'ютерних систем, що можуть призводити до збоїв як програмного забезпечення, так і комп'ютерної системи в цілому.

Відмова – тип помилки, або сукупності помилок, що призводить до переходу системи від нормального стану функціонування до непрацездатного стану. Відмова супроводжується відхиленням поведінки програми від визначених у специфікації вимог та обмежень, що обумовлено виникненням прихованих дефектів проектування або збоїв у середовищі функціонування [4, 8].

Основними причинами відмов є:

– невиконання вимог специфікації або помилкове її трактування, що породжує її помилкове трактування і не відповідає очікуванням користувача;

– невідповідність вимоги, наведеної у специфікації програмного забезпечення, до вимог середовища виконання або апаратного забезпечення;

– наявність помилок у програмі, що, як приклад, може призводити до несанкціонованого доступу до інформації або компонентів системи;

– неправильність, неточність або незавершеність реалізації алгоритмів у програмному коді.

Отже, до відмов можуть призводити як одиничні помилки у програмному коді, так і їх сукупність, а також невиявлені дефекти проектування та програмування.

Згідно [14], помилки програмного забезпечення поділяють на такі класи:

– помилки виконання функцій та логічні помилки;

– помилки, пов’язані з результатами обчислень і продуктивністю;

– помилки вводу, опрацювання і відображення інформації;

– помилки інтерфейсів;

– помилки, пов’язані з інтерпретацією даних.

Логічні помилки виникають у результаті порушення принципів функціонування тих чи інших алгоритмів, неузгодженості змінних чи операторів, а також правил і стилів програмування [5].

До появи функціональних помилок призводять неточності при визначенні функцій, неправильний порядок їх виклику, відсутність або недосконаліть функцій обробки нестандартних подій та інші.

Помилки, пов’язані з результатами обчислень, виникають через некоректність вхідних даних, помилки у розрахунках, суттєві похибки методів обчислень.

Помилки щодо продуктивності роботи програмного забезпечення, пов’язані із застосуванням повільних алгоритмів опрацювання даних і супроводжуються високим часом виконання запитів, низькою продуктивністю відгуку системи та проблемою відновлення працездатності системи на заданому рівні.

Причинами, що обумовлюють помилки вводу, опрацювання і відображення інформації є недосконалі методи представлення даних при їх зчитуванні або записі у базу даних.

Помилки інтерфейсу проявляються при некоректному зверненні один програмних модулів до інших, одержанні та передачі даних у непідтримуваних

форматах, некоректній інтерпретації даних середовищем виконання програмного забезпечення.

Помилки, пов’язані з об’ємом опрацювання інформації, виникають у зв’язку з недостатністю апаратних ресурсів, відсутністю або недосконалістю механізмів масштабування операцій маніпулювання даними, високою інтенсивністю запитів до програмного забезпечення.

Наведені класи помилок програмного забезпечення у різному програмному забезпеченні проявляються по-різному. Зокрема, для комп’ютерних систем, в основі якого лежить функціонування програмного забезпечення і баз даних, характерна концентрація помилок вводу та опрацювання даних, логічних помилок та помилок продуктивності. Для високонавантажених обчислювальних комп’ютерних систем характерні помилки обчислень, а для інформаційно-управляючих систем – помилки функціонування та логічні помилки.

Для виявлення різних типів помилок, дефектів та відмов необхідно проектувати плани перевірок і застосування різних видів тестування програмного забезпечення, що дає змогу значно знизити імовірність їх появи на етапі впровадження та супроводу.

Підхід до класифікації помилок програмного забезпечення запропоновано фірмою IBM, який називають ортогональною класифікацією дефектів [8]. Даний підхід передбачає, що до функцій відповідального за тестування розробника, входить власне визначення дефектів програмного забезпечення і їх групування за визначеними класами. Класифікація дефектів не залежить від типу програмного забезпечення, організації процесу розробки і може застосовуватись як універсальна класифікація. Згідно ортогональної класифікації дефектів, розрізняють класи, які представлено у табл. 1.1 [5].

Таблиця 1.1.

Класи дефектів за ортогональною класифікацією IBM

Клас помилки	Опис дефекту
Функція	Помилки щодо відображення даних у інтерфейсах користувачів ПЗ, які пов’язані зі збоями апаратного забезпечення або зовнішніми структурами даних.
Інтерфейс	Помилки методів і засобів інтеграції програмних модулів або систем, механізмів їхньої взаємодії.
Логіка	Логічні помилки у програмно реалізованих алгоритмах опрацювання даних, які не були перевірені, а також неправильна інтерпретація значень змінних .
Присвоювання	Помилки ініціалізації змінних, або не коректне застосування структур представлення даних.
Зациклення	Помилки, пов’язані з продуктивністю програмного забезпечення, які можуть бути викликані неоптимальним розподілом ресурсів.
Середовище	Помилки в репозиторії, у керуванні змінами або в контрольованих версіях проекту
Алгоритм	Помилки, що викликані неефективними алгоритмами опрацювання даних, некоректними структурами даних або помилками в алгоритмах.
Документація	Помилки у документації процесів розробки програмного забезпечення на різних стадіях життєвого циклу

Для ортогональної схеми класифікації не є характерним мультикласифікація однієї і тієї ж помилки. Це означає, що будь-яке твердження про помилку відноситься тільки до однієї категорії і як наслідок, помилка у програмному забезпеченні повинна знаходитись тільки в одному програмному

класі. Це забезпечує однозначність та уніфікованість дій розробників при виявленні дефектів або інших типів помилок.

Іншим підходом до класифікації дефектів програмного забезпечення є підхід, запропонований фірмою Hewlett–Packard. В основі цього підходу лежить класифікація Буча, що дає змогу встановити співвідношення між різними типами помилок на стадіях життєвого циклу (рис. 1.2).

Рис. 1.2. Співвідношення між типами помилок ПЗ

Таке співвідношення помилок характерне для багатьох фірм-розробників, однак існують деякі відхилення між іншими розробниками програмного забезпечення комп’ютерних систем.

У відповідності до [9] вартість етапу специфікації вимог і внесення змін у нього, оцінюється приблизно у 10% вартості проекту.

Щодо кодування, то його вартість становить трохи більше, ніж 20%, а тестування – 45% загальної вартості проекту. Окрім цього, значний обсяг ресурсів використовують при супроводі програмного забезпечення і виправлення дефектів, що проявилися.

Як показує практика, чим пізніше виявляється дефект програмного забезпечення, тим дорожче вартість його виправлення, яка приблизно відповідає експонентному закону розподілу.

До відмови програмного забезпечення призводять помилки, що відносяться до одного з наведених вище класів, а відповідно це впливає на надійність функціонування комп'ютерної системи. Для виявлення джерела помилки та усунення дефектів програмного забезпечення, що впливає на надійність комп'ютерної системи, можливе застосування наступного алгоритму:

– виявлення помилок у технологіях розробки програмного забезпечення;

– ідентифікація взаємозалежності помилок при проектуванні програмного забезпечення, викликаних людським фактором;

– формування списку відмов, помилок і дефектів згідно визначених класів на етапах життєвого циклу;

– порівняння помилок, спричинених людським фактором на етапах розробки програмного забезпечення комп'ютерних систем і дефектів, пов'язаних з помилками у специфікації і використовуваних моделях програмного забезпечення;

– застосування методів і засобів перевірки помилок на всіх етапах розробки програмного забезпечення;

– визначення залежності між дефектами програмного забезпечення і відмовами комп'ютерної системи з метою розробки методів ідентифікації, збору та аналізу факторів впливу на надійність комп'ютерних систем;

– формування і впровадження методик документування і супроводу програмного забезпечення комп'ютерних систем.

Такий підхід до фіксації взаємозалежності між дефектами і відмовами має назву причино-наслідкових зв'язків «помилка-відмова». Такий підхід дозволяє забезпечити підбір ефективних засобів тестування і моніторингу стану програмного забезпечення і відповідно зменшити негативний вплив збоїв програмного забезпечення на надійність комп'ютерних систем.

При цьому можливе використання наступної класифікації типів відмов програмного забезпечення комп’ютерних систем:

– апаратна відмова, що зумовлює непрацездатність системного програмного забезпечення;

– інформаційна відмова – тип відмови, що зумовлена помилками вводу даних або методами і засобами передачі даних;

– програмна відмова – характеризується впливом дефектів у програмних компонентах комп’ютерної системи;

– ергономічна відмова – відмова, що викликана людським фактором при взаємодії користувача з компонентами комп’ютерної системи: апаратною або програмною.

Помилки програмного забезпечення можуть бути спричинені різними факторами і на різних стадіях розробки програмного забезпечення комп’ютерних систем, тому необхідно проводити додаткові дослідження щодо виявлення помилок і дефектів ПЗ та розробки процедур визначення впливу збоїв на надійність комп’ютерних систем.

1.3. Аналіз моделей надійності комп’ютерних систем і програмного забезпечення

Для опису надійності апаратної складової комп’ютерних систем можна використовувати моделі надійності технічних засобів, однак, для опису надійності програмного забезпечення такі моделі не можна застосовувати, зважаючи на різний механізм і природу відмов.

Основна відмінність апаратного і програмного забезпечення полягає у складності зв’язків між компонентами ПЗ та відсутністю фізичного старіння. Відмови програмного забезпечення залежать лише від вмісту і кількості дефектів, а також від характеристик середовища експлуатації.

За результатами експериментів встановлено, що інтенсивність відмов програмного забезпечення зменшується в процесі виявлення та усунення

дефектів, а стабільна робота програмного забезпечення починається приблизно після чотирьох років експлуатації [8]. На рис. 1.3 наведено графіки інтенсивності відмов апаратного та програмного забезпечення у часі.

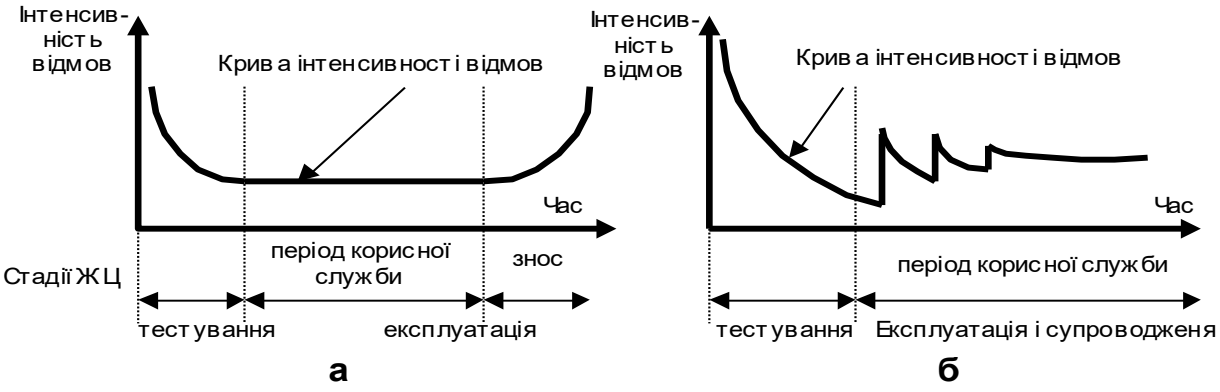

Рис. 1.3. Інтенсивність відмов апаратного та програмного забезпечення

Дефекти, що призводять до збоїв програмного забезпечення пов’язані з помилками розробників на різних етапах створення продукту. На рис. 1.4 наведено криву дефектів ПЗ за стадіями його розробки на основі водоспадної моделі життєвого циклу.

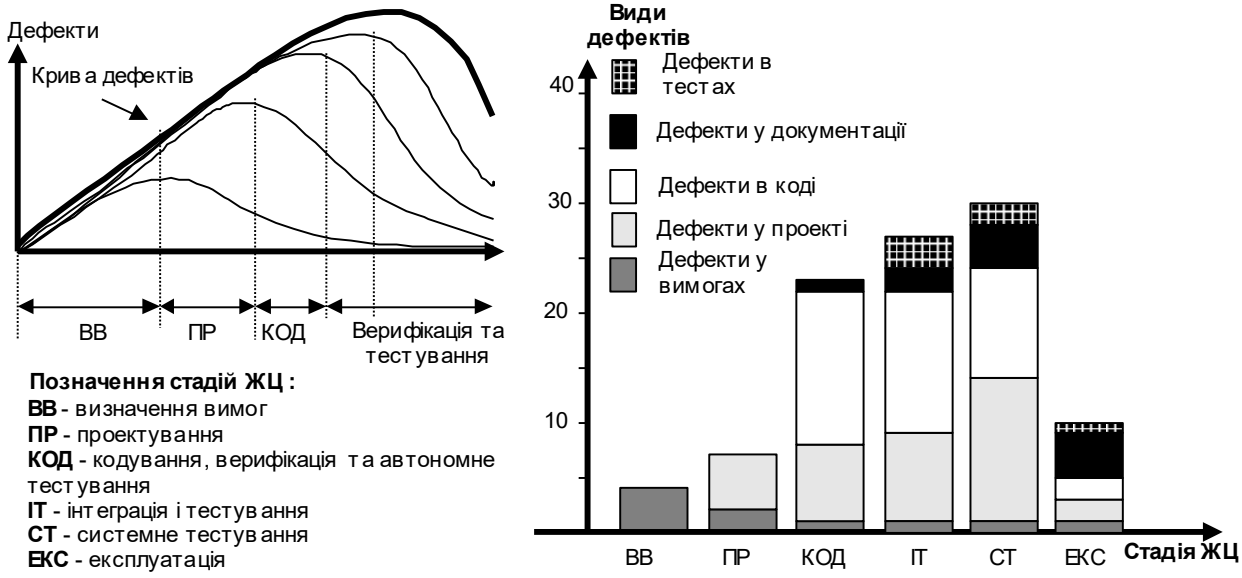

Рис. 1.4. Крива дефектів програмного забезпечення за етапами водоспадної моделі життєвого циклу

Взаємозв’язок між відмовами, які спричиняють збої програмного забезпечення, дефектами та помилками у ньому, викликані різною етіологією показано на рис. 1.5.

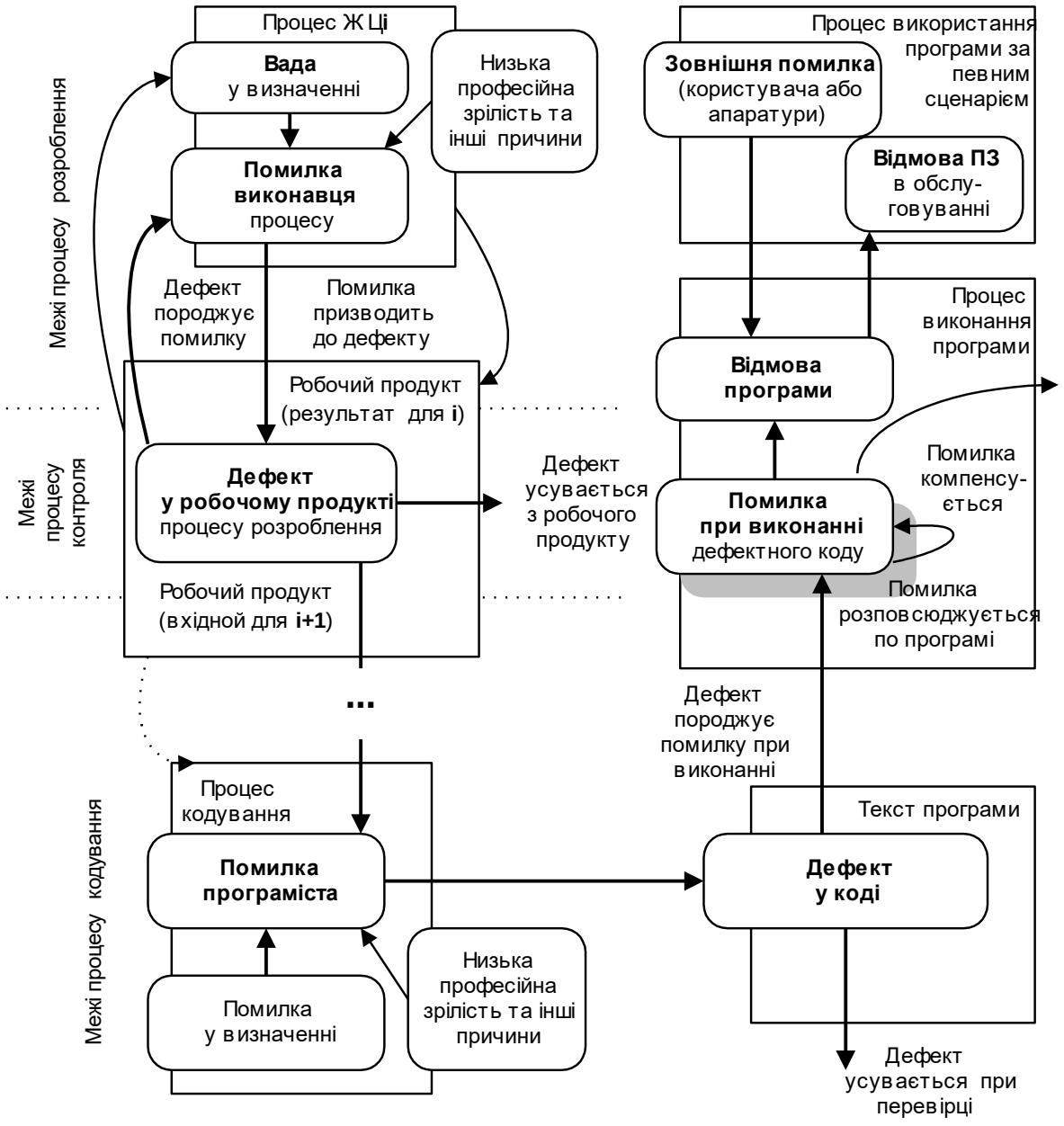

Рис. 1.5. Взаємозв’язок між відмовами, помилками і дефектами програмного забезпечення

Якщо у результаті інспекцій програмного коду, модульного та інтеграційного тестування, залишились невиявленими дефекти, то вони можуть проявитись пізніше під час функціонування комп’ютерної системи. Якщо дефект

програмного забезпечення проявляє себе на етапі експлуатації, то може виникнути ланцюгова реакція щодо помилковості функціонування комп'ютерної системи, яке може привести до зниження надійності та відмовостійкості системи в цілому.

Тому важливою сферою розробки комп'ютерних систем, і програмного забезпечення зокрема, є забезпечення заданого рівня надійності програмного забезпечення і зниження негативного впливу дефектів на надійність комп'ютерної системи під час її експлуатації. У роботі [5] наведено задачі, розв'язок яких позитивно впливає на покращення надійності програмного забезпечення, зокрема:

– визначення факторів впливу на надійність та обґрунтування вимог до надійності програмного забезпечення на конкретній стадії життєвого циклу;

– визначення стратегії і методів забезпечення виконання вимог на кожному етапі створення комп'ютерної системи;

– ідентифікація об'єктів перевірки та розробка процедур кількісного вираження їх властивостей для прийняття рішень щодо їх оптимізації.

Виходячи з результатів проведеного аналізу, можна зробити висновок про те, що найбільш важливою характеристикою надійності програмного забезпечення є завершеність. На основі кількісних показників завершеності можна визначити вплив дефектів програмного забезпечення на надійність комп'ютерної системи.

Сьогодні вченими побудовано та обґрунтовано багато моделей опису та оцінювання характеристики завершеність програмного забезпечення. Однак, все ж залишаються відкритими ряд питань щодо показників кількісного їх вираження, обґрунтованості та адекватності результатів оцінювання на етапах життєвого циклу та впливу збоїв програмного забезпечення на надійність комп'ютерної системи.

Характеристика завершеність може описуватись різними критеріями в залежності від типу комп'ютерної системи, частоти використання окремих її

компонентів, продуктивності та вартісних показників забезпечення завершеності програмного забезпечення.

Одним з підходів до визначення завершеності програмного забезпечення є не критеріальний підхід, суть якого полягає у призначенні однакового цільового значення надійності усім компонентам комп'ютерної системи без розподілу. Не критеріальний підхід доцільно використовувати в умовах невизначеності, а у випадку наявності документації та доступу до програмного коду, даний підхід є неефективним, оскільки встановлює необґрунтовані вимоги до надійності. При висуванні занадто високих вимог до надійності програмного забезпечення, його реалізація може вимагати значених часових і фінансових ресурсів, або бути недосяжною [7].

Технічні підходи [8, 9] передбачають побудову блок-схем надійності програмно-апаратних комплексів комп'ютерних систем або рівномірне пропорційне розподілення надійності за компонентами з врахуванням послідовної і паралельної їх роботи. Технічні підходи породжені методами і моделями теорії надійності технічних засобів і прийнятні лише для вбудованого програмного забезпечення за умови наявності значень надійності для усіх компонентів комп'ютерної системи.

Підходи на основі специфікації використання програмних систем володіють найбільш широким спектром методів і засобів виявлення та оцінювання дефектів програмного забезпечення і відповідно визначення їх надійності на різних рівнях.

Одним з перших підходів був підхід [10], що передбачав розподіл надійності з урахуванням ймовірностей міжкомпонентних переходів в алгоритмах роботи програмного забезпечення і дав поштовх до подальшого розвитку досліджень надійності.

Інший підхід [11] передбачає розподіл надійності за функціональним або операційним профілем з використанням ієрархічної структури та врахуванням особливостей груп користувачів, режимів роботи системи, функцій та операцій. Перевагами використання такого підходу є можливість враховувати особливості

масштабних комп’ютерних систем, будувати сценарії тестування, які враховують специфіку середовища використання комп’ютерної системи та заощаджувати час на тестування програмної складової комп’ютерної системи. До недоліків системи варто віднести чутливість моделей до змін операційного профілю.

Модель [12] заснована на розподіленні надійності у виконавчому профілі програмного забезпечення та використовує моделювання функцій стохастичним процесом запуску та відпрацювання модулів, які їх реалізують. У результаті аналізу переходів між програмними компонентами можна побудувати шаблони виконання кожної функції програмного забезпечення комп’ютерних систем. Недоліком такого підходу є необхідність інтеграції сервісного програмного коду для визначення частоти звернення до модуля чи компонента.

Моделі і методи [13,14] передбачають використання марківських моделей передачі керування між компонентами програмного забезпечення та переходу з одного стану в інший. Перевагою моделі [13] є те, що вона враховує погляд розробників на можливі сценарії роботи системи, а [14] – враховує погляд користувачів комп’ютерної системи. До недоліків обох моделей належить складність побудови моделей для масштабних комп’ютерних систем.

Розвитком наведених вище моделей є підхід [15], що використовує розподіл надійності на основі ієрархічних моделей станів. До складу моделей входять ієрархічна складова, що представляється у вигляді дерев, та поведінкова – описується ланцюгами Маркова. Перевагою [15] є те, що модель враховує погляд користувачів щодо виконання функцій комп’ютерною системою, а недолік полягає у невизначеності простору станів системи та складністю моделювання систем великого розміру.

Ще однією моделлю, що входить до технічних підходів визначення надійності на основі специфікації використання програмного забезпечення є модель [16]. Дана модель передбачає розподілення надійності за ознаками операційної критичності та режимами відмов і базується на методі Software

Failure Mode and Effect Analysis. Така модель дозволяє врахувати аспекти комп'ютерних систем, які змодельовані засобами мови UML.

Модель, запропонована Н. Олсоном, входить до підкласу технічних підходів, що враховує розподіл надійності з врахуванням властивостей компонентів ПЗ, зокрема оцінювання складності модулів та схильності їх до помилок. Недоліком моделі є ігнорування погляду користувачів на сценарії виконання функцій комп'ютерної системи.

Розроблено також моделі, які використовують розподіл надійності з врахуванням ресурсів проектів. При цьому, моделі М. Хелендера та М. Ліу [17, 18], описує розподіл надійності програмного забезпечення з врахуванням часу тестування шляхом застосування методів розв'язку задач нелінійної оптимізації – мінімізації часу тестування і потребує застосування спеціалізованих математичних пакетів. Розподілення надійності на основі операційного профілю ПС з урахуванням витрат на досягнення цілей надійності.

Модель [17] є придатною для визначення комплексного розподілу надійності при заданому цільовому значенні інтенсивності відмов для програмного забезпечення, що взаємодіє з багатьма іншими засобами.

Модель [18] поєднує два підходи до планування надійності та вартості розробки: мінімізація вартості за умови обмеження надійності та максимізація надійності за умови обмеження бюджету.

На основі результатів аналізу моделей розподілу надійності програмного забезпечення і виявлення впливу дефектів та збоїв ПЗ на надійність комп'ютерних систем можна визначити наступні вимоги до моделі розподілу надійності за компонентами програмного забезпечення:

– врахування поглядів користувачів комп'ютерних систем на важливість сценаріїв експлуатації та виклику функцій у загальному бізнес-процесі;

– врахування поглядів розробників програмного забезпечення на важливість та надійність виконання функцій комп'ютерною системою;

– врахування інтенсивності звернень та значень надійності окремих компонентів програмного забезпечення комп’ютерної системи;

– забезпечення максимального рівня задоволеності надійністю функціонування компонентів програмного забезпечення комп’ютерної системи, що відповідають очікуванням користувачів.

1.4. Аналіз методів прогнозування надійності програмного забезпечення комп’ютерних систем

Важливим процесом при розробці комп’ютерних систем є прогнозування надійності програмного забезпечення, як основи управління даними і процесами в заданому середовищі використання. Побудова моделей надійності, зокрема на ранніх етапах створення комп’ютерних систем, обумовлена необхідністю оцінювання ресурсів для досягнення мети розробки системи. Результати оцінювання дають змогу оптимально розподіляти ресурси за стадіями розробки на основі відхилень від плану, ефективно керувати самим процесом розробки для досягнення необхідно рівня надійності програмного забезпечення і комп’ютерної системи в цілому.

Як показує практика, прогнозування надійності необхідно проводити як на ранніх етапах проектування, так і на пізніх. Для цього можна використати комбінацію моделей раннього та пізнього прогнозування із заданою ітераційністю. Це дає змогу забезпечити уточнення параметрів надійності та забезпечити моніторинг за розвитком і усуненням дефектів на усіх етапах життєвого циклу.

У результаті прогнозування надійності одержують наступні дані:

– очікувана кількість дефектів на стадіях розробки комп’ютерної системи, зокрема програмного забезпечення;

– очікувана інтенсивність відмов за компонентами програмного забезпечення;

– варіанти оптимізації програмного забезпечення і процесів розробки.

В загальному випадку, алгоритм процесу прогнозування надійності представлено на рис. 1.6.

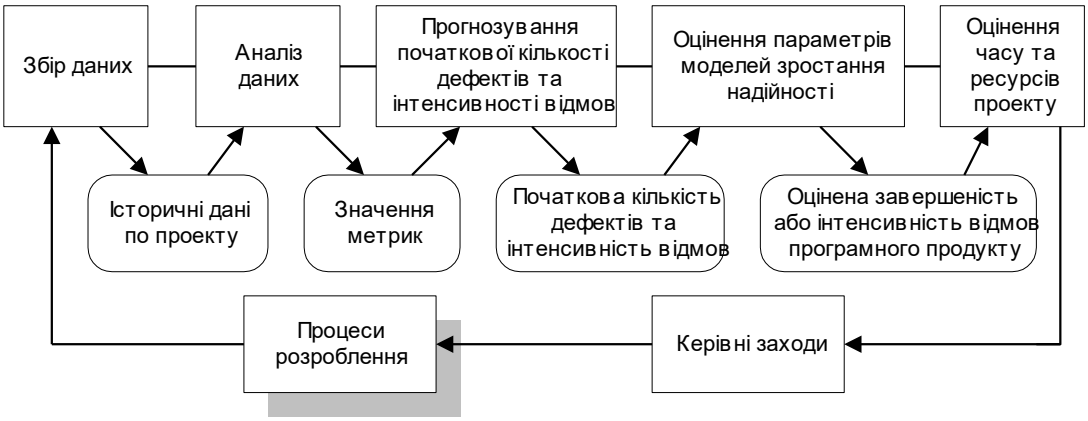

Рис. 1.6. Процес прогнозування надійності ПС

У процесі розробки програмного забезпечення комп'ютерних систем кількісні значення вимірюваних величин змінюються і на етапі тестування проводиться заміна спрогнозованих значень надійності на фактичні. Це дає змогу визначити статистичні параметри оцінювання зростання надійності протягом життєвого циклу розробки комп'ютерної системи.

До переваг оцінювання показників за моделлю зростання надійності належать наступні:

– вхідні тестові дані, що є малими за розміром, можуть провокувати нестабільні або неможливі значення параметрів надійності програмного забезпечення. У такому випадку значення, одержані за допомогою ранніх моделей прогнозування надійності, можна використати для порівняння або стабілізації прогнозів шляхом доповнення даних, одержаних через тестування;

– при використанні ітераційних технологій оцінювання значень показників надійності можливе виникнення залежності від початкових оцінок і при використанні апріорних значень як базової початкової оцінки звужується пошук наближених значень параметрів моделі надійності;

– використання історичних даних, як апріорних значень надійності дає змогу забезпечити виконання процесу планування та розподілу ресурсів на етапі тестування програмного забезпечення комп’ютерних систем.

У результаті виконання кожного етапу зі створення програмного забезпечення комп’ютерних систем формується звіт щодо оцінки показників надійності на основі якого визначається міра досягнення поставлених цілей, оптимальність використання ресурсів, недоліки процесів розробки та шляхи їх вдосконалення.

Якщо прогнозовані оцінки надійності не відповідають визначеним вимогам, то це є сигналом до втручання та реконструкції процесу розробки програмного забезпечення комп’ютерних систем.

На основі застосування моделей прогнозування надійності програмного забезпечення можна забезпечити процес керування розробкою та супроводом комп’ютерних систем із врахуванням кількісних критеріїв та визначати потенційний вплив дефектів або збоїв програмного забезпечення на функціонування комп’ютерної системи. В комплексі з іншими моделями прогнозування можна забезпечити ефективний спосіб керування ризиками і тим самим збалансувати проект за тривалістю, вартістю і показниками надійності експлуатації комп’ютерних систем.

1.5. Висновки

У даному розділі проведено аналіз принципів організації комп’ютерних систем, досліджено базові структури та компоненти, проаналізовано типи помилок програмного забезпечення та їх класифікацію. Окрім цього, проведено аналіз моделей надійності комп’ютерних систем з різних точок зору та методів прогнозування їх надійності.

Основні наукові та практичні результати, які досягнуто у даному розділі:

1. У результаті аналізу базових принципів організації та побудови комп’ютерних систем, встановлено, що одним з найбільш важливих її

компонентів є програмне забезпечення, від стабільності і надійності роботи якого залежить ефективність експлуатації комп’ютерної системи.

2. Проведено аналіз помилок програмного забезпечення, їх типів та впливу на надійність, що дало змогу враховувати їх розвиток у комп’ютерній системі та обґрунтувати необхідність застосування моделей прогнозування показників надійності на різних стадіях життєвого циклу і спостереження за ними під час експлуатації і супроводу.

3. Проведено аналіз моделей надійності комп’ютерних систем з акцентом на програмне забезпечення і виявлено, що застосування моделей оцінювання надійності технічних засобів є не ефективним для представлення показників надійності програмного забезпечення у зв’язку з різною природою та механізмами відмов.

4. На основі результатів аналізу моделей раннього і пізнього прогнозування помилок програмного забезпечення обґрунтовано їх застосування при проектуванні комп’ютерних систем, що дає змогу побудувати комплексний підхід до визначення впливу дефектів програмного забезпечення на надійність експлуатації комп’ютерних систем з врахуванням історії виникнення помилок у ПЗ.

РОЗДІЛ 2
ПОБУДОВА МОДЕЛІ ТА РОЗРОБКА МЕТОДУ АНАЛІЗУ ВПЛИВУ ПОМИЛОК ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА НАДІЙНІСТЬ КОМП'ЮТЕРНИХ СИСТЕМ

2.1. Формалізація критеріїв надійності програмних складових комп'ютерних систем

Для визначення показників надійності програмного забезпечення комп'ютерних систем скористаємось нотаціями і означеннями, які рекомендовано у стандарті [2]. Згідно стандарту [2] надійність програмного забезпечення описується характеристиками завершеності, відмовостійкості та відновлюваності, як показано на рис. 2.1.

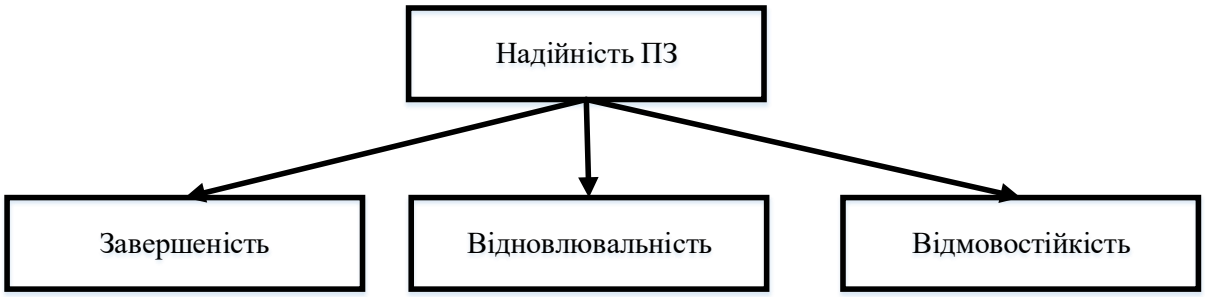

Рис. 2.1. Характеристики надійності програмного забезпечення комп'ютерних систем

Характеристика завершеність є однією з найбільш значущих характеристик надійності програмного забезпечення, в контексті впливу на комп'ютерну систему, оскільки описує безвідмовність роботи ПЗ і відображає ті властивості, що забезпечують уникнення збою через приховані чи невиявлені дефекти.

Для кількісного представлення надійності, зокрема, завершеності програмного забезпечення, використовуються метрики, що згідно [2], представляють собою "модель вимірювання атрибута, яка пов'язана з

характеристикою надійності ПЗ і представляє собою комбінацію конкретного методу вимірювання атрибута сутності і шкали вимірювання”.

В залежності від етапу розробки програмного забезпечення можна використовувати набори внутрішніх, зовнішніх та експлуатаційних метрик.

Внутрішні метрики використовують для вимірювання показників надійності кожного окремо взятого програмного модуля, зовнішні – при функціонуванні програмного забезпечення в середовищі розробників, а експлуатаційні – при виконанні функцій програмним забезпеченням у реальному середовищі виконання.

З точки зору забезпечення надійності комп’ютерних існує залежність між показниками надійності на внутрішньому, зовнішньому та експлуатаційному рівні. Залежність між атрибутами надійності на трьох рівнях можна представити у вигляді, як показано на рис. 2.2.

Рис. 2.2. Представлення надійності програмного забезпечення

В якості метрик завершеності програмного забезпечення на рівні програмних модулів пропонується використовувати наступні:

– інтенсивність відмов;
– інтенсивність усунення дефектів;
– точність перевірок.

Кількісно інтенсивність виявлення дефектів виражається як відношення кількості виявлених дефектів до очікуваної кількості дефектів

$$Def^{int} = \frac{Def^{find}}{Def^{pend}} \quad (2.1)$$

Def^{int} – метрика інтенсивності виявлення дефектів;

Def^{find} – кількість виявлених дефектів;

Def^{pend} – кількість очікуваних дефектів.

Інтенсивність усунення дефектів можна визначити як кількість виявлених та усунених дефектів або як відношення кількості усунутих дефектів до загальної кількості виявлених

$$Def^{likv} = n \quad (2.2)$$

Def^{likv} – інтенсивність усунення дефектів;

n – кількість усунутих дефектів

У випадку, коли кількість виявлених та кількість усунутих дефектів однакова, отримаємо

$$Def^{find} = Def^{likv} \quad (2.3)$$

Інша метрика, що описує інтенсивність усунення дефектів на внутрішньому рівні представляється як

$$Def^{likv} = \frac{n}{Def^{find}} \quad (2.4)$$

Інтерпретації даної метрики така, що чим ближче Def^{likv} до 1, тим менше дефектів залишилось у програмних модулях.

Точність перевірок можна обчислити як відношення кількості категорій дефектів, що заплановані до перевірки, до кількості категорій дефектів, які необхідно охопити для забезпечення повноти і точності перевірки

$$Check^{accur} = \frac{Cat^{plan}}{Cat^{need}} \tag{2.5}$$

$Check^{accur}$ – точність перевірок;

Cat^{plan} – кількість категорій дефектів запланованих до перевірки;

Cat^{plan} – кількість перевірок, необхідних для забезпечення повноти виявлення дефектів.

До зовнішніх метрик характеристики завершеність, як основної характеристики надійності програмного забезпечення належить:

- щільність прихованих дефектів;
- щільність відмов;
- щільність дефектів;
- інтенсивність усунення дефектів;
- середній час між відмовами.

Щільність прихованих дефектів можна визначити як відношення різниці між загальною кількістю дефектів у програмному продукті та загальною кількістю виявлених дефектів до розміру програмного продукту (кількості програмних компонентів)

$$Density^{Def^{Pend}} = \frac{|Def^{pend} - Def^{find}|}{V} \tag{2.6}$$

$Density^{Def^{Pend}}$ – щільність прихованих дефектів;

Def^{pend} – кількість очікуваних дефектів;

Def^{find} – кількість реально виявлених дефектів;

V – кількість компонентів програмного забезпечення (розмір ПЗ).

Щільність відмов описується відношенням кількості відмов за певний період до кількості виконаних тестів

$$Density^{Failure} = \frac{|Failure|}{|Tests|} \tag{2.7}$$

$Density^{Failure}$ – щільність відмов за тестовий період;

$|Failure|$ – кількість відмов за тестовий період;

$|Tests|$ – кількість виконаних тестів.

Щільність дефектів на зовнішньому рівні вимірюється як відношення загальної кількості виявлених дефектів або відмов у програмному продукті до його розміру

$$Density^{Def} = \frac{|Def^{find}|}{V} \tag{2.8}$$

$Density^{Def}$ – щільність дефектів;

Def^{find} – кількість виявлених дефектів;

V – кількість компонентів програмного забезпечення (розмір ПЗ).

Інтенсивність виявлення дефектів на зовнішньому рівні обчислюється як відношення кількості виправлених дефектів до кількості виявлених дефектів

$$Def_{int}^{ext} = \frac{|Def^{change}|}{|Def^{find}|} \tag{2.9}$$

Def_{int}^{ext} – інтенсивність дефектів на зовнішньому рівні;

$|Def^{change}|$ – кількість виправлених дефектів;

$|Def^{find}|$ – кількість виявлених дефектів.

Для визначення середнього часу між відмовами можна скористатись відношеннями періоду часу функціонування програмного забезпечення до кількості виявлених відмов або суми часу безперебійної роботи до кількості виявлених дефектів за даний період

$$Avg_{time}^{Failure} = \frac{T^{func}}{|Failure|}$$
$$Avg_{time}^{Failure} = \frac{\sum T^{func}}{|Failure|} \quad (2.10)$$

$Avg_{time}^{Failure}$ – середній час між відмовами;

$|Failure|$ – кількість відмов за деякий період часу;

T^{func} – час перебування програмного забезпечення у працездатному стані.

Для кількісного вираження завершеності програмного забезпечення на експлуатаційному рівні можна використовувати експертні технології. Однак, з метою визначення впливу дефектів і відмов, які призводять до збоїв програмного забезпечення, на надійність експлуатації комп'ютерної системи необхідно побудувати та формалізувати відповідну модель.

2.2. Побудова моделі визначення впливу збоїв програмного забезпечення на надійність комп'ютерних систем

Як було зазначено у першому розділі дипломної роботи магістра, до складу типової комп'ютерної системи входять: апаратна складова, програмне забезпечення різних рівнів та канали передачі даних, тому представимо комп'ютерну систему у вигляді сукупності:

$$CompSyst=\{HW,SW,Chan\} \quad (2.11)$$

CompSyst - комп'ютерна система;

HW – сукупність апаратного забезпечення;

SW – сукупність програмного забезпечення;

Chan – сукупність каналів передачі даних.

В загальному випадку, надійність комп'ютерної системи залежить від надійності її компонентів і виражається як зважена сума показників надійності компонентів формули (2.11) .

Для представлення надійності програмного забезпечення можна записати наступний вираз

$$R(SW)=\{SW_i, Def_{ij}\} \quad (2.12)$$

$R(SW)$ – надійність програмного забезпечення комп'ютерної системи;

SW_i – компонент програмного забезпечення, $i=1..k$, k – кількість компонентів програмного забезпечення;

Def_{ij} – дефект, пов'язаний з функціонуванням *i*-го компоненту програмного забезпечення, $j=1..m$, m – кількість дефектів.

$$R(HW)=\{HW_i, Def_{ij}\} \quad (2.13)$$

$R(HW)$ – надійність апаратного забезпечення комп'ютерної системи;

HW_i – компонент апаратного забезпечення, $i=1..l$, l – кількість компонентів апаратного забезпечення;

Def_{ij} – дефект, пов'язаний з функціонуванням *i*-го компоненту апаратного забезпечення, $j=1..p$, p – кількість дефектів.

З іншого боку при проектуванні архітектури комп'ютерної системи, в тому числі й архітектури програмного забезпечення, встановлено зв'язки між компонентами чи модулями системи, які представляють метрику зчеплення між

модулями. Для представлення зв'язків між компонентами програмного забезпечення можна записати

$$Rel_{sw} = \{ SW_i, SW_j \} \tag{2.14}$$

SW_i – i-ий компонент програмного забезпечення комп'ютерної системи;

SW_j – j-ий компонент програмного забезпечення комп'ютерної системи, $i, j \in K, i \neq j$.

Для апаратного забезпечення комп'ютерної системи зв'язки між ними можна представити по аналогії до (2.14)

$$Rel_{HW} = \{ HW_i, HW_j \} \tag{2.15}$$

HW_i – i-ий компонент апаратного забезпечення комп'ютерної системи;

HW_j – j-ий компонент апаратного забезпечення комп'ютерної системи, $i, j \in L, i \neq j$.

Для представлення зв'язків між апаратними компонентами комп'ютерної системи і програмними складовими запишемо

$$Rel_{HW}^{SW} = \{ \{ SW_i, HW_j \}, Chan_s \} \tag{2.16}$$

Rel_{HW}^{SW} – множина зв'язків між програмним і апаратним забезпеченням комп'ютерної системи;

$\{ SW_i, HW_j \}$ – пара «програмне забезпечення-апаратний пристрій», які взаємодіють між собою, $i \in K, j \in L$;

$Chan_s$ – один з каналів обміну даними між програмним і апаратним забезпеченням, $s = 1..S$, S – кількість каналів передачі даних.

Таким чином, за допомогою побудови ланцюга взаємодії програмного і апаратного забезпечення комп'ютерних систем, можна визначити потенційний вплив дефектів програмного забезпечення, що призводять до збоїв, на надійність комп'ютерної системи, яка описуються такими ж атрибутами надійності що й програмне забезпечення.

Для підвищення надійності функціонування комп'ютерних систем пропонується мінімізувати зчеплення між модулями як всередині програмного забезпечення, так і зв'язків між програмними і апаратними складовими, шляхом використання підходів функційного програмування.

Наступний етап дослідження полягає в обґрунтуванні моделі представлення дефектів та розробці методу спостереження за розвитком дефектів програмного забезпечення.

2.3. Обґрунтування моделі представлення дефекту програмного забезпечення комп'ютерних систем

Модель дефекту програмного забезпечення повинна описувати елемент програмного забезпечення у якому виявлено дефект.

Нехай $SW=\{sw_1,sw_2,sw_3,...,sw_k\}$ – множина програмних елементів у програмному забезпеченні комп'ютерної системи. Сукупність дефектів або помилок у конструкціях програмного забезпечення представимо у вигляді множини $Def=\{def_1,def_2,def_3,...,def_p\}$. Виходячи з цього, модель дефекту можна представити кортежем функцій [20]

$$<Func_{Def},Avg_{Def}> \qquad (2.17)$$

$Func_{Def}$ – функція, що описує розвиток дефекту;

Avg_{Def} – середня інтенсивність проявів властивостей дефекту.

Для того, щоб побудувати модель дефектів програмного забезпечення комп'ютерної системи пропонується алгоритм, що передбачає виконання п'яти кроків:

1. Визначити та сформулювати правило опису синтаксичних конструкцій компонентів програмного забезпечення. Сукупність $Rules = \{r_1, r_2, r_3, ..., r_p\}$ формує набір правил і визначається у відповідності до документації мов, технологій та підходів програмування. Дефект синтаксичної конструкції представляється як порушення правила. Якщо $def \in Def$ – дефект програмного забезпечення і $r \in Rules$ – правило, якому повинна відповідати конструкція, то $def = violate(r)$ означає порушення правила r спричинене дефектом def [20].

2. Визначити ознаки $S_d = \{s_{d,1}, s_{d,2}, s_{d,3}, ..., s_{d,m}\}$ порушення правила за допомогою засобів аналізу синтаксичних та логічних конструкцій з яких побудовано програмне забезпечення.

3. Обґрунтувати застосування метрики для встановлення інтенсивності елементарних властивостей дефекту програмного забезпечення комп'ютерної системи. Властивість (ознака) дефекту може бути елементарною або формувати складену з набору декількох властивостей. Характерною особливістю елементарних властивостей дефекту є те, що їх значення можна виміряти за допомогою лише однієї метрики [20]. Для будь-якої елементарної властивості дефекту $s \in S$ обирається метрика $M_s : E \to Q$, Q – область допустимих значень при кількісному вираженні властивості дефекту S.

4. Встановити граничний поріг для метрик. Порогове значення $Z_s \in Q$ забезпечує можливість трактування результату вимірювання властивості дефекту щодо її розвитку, в залежності до якої області вона належить.

5. Побудувати $Func_{def}$ та Avg_{def}.

При визначенні оцінки інтенсивності елементарної властивості дефекту $s \in S$, можна використати функції, що враховують тип порогового значення: метрики: верхній або нижній [20]:

– у випадку, якщо граничне значення представляє верхню межу метрики:

$$I_s(sw) = HigherThan(M_s(sw), Z_s) = \frac{M_s(sw)}{Z_s} \cdot 100 \quad (2.18)$$

– у випадку, якщо граничне значення представляє нижню межу метрики:

$$I_s(sw) = LowerThan(M_s(sw), Z_s) = \frac{Z_s}{M_s(sw)} \cdot 100 \quad (2.19)$$

При визначенні інтенсивності складених властивостей дефектів програмного забезпечення комп'ютерних систем можливе застосування функцій агрегації:

– при одночасному прояві елементарних властивостей дефекту $s_1, s_2 \in S$, які формують складену властивість *S* у [20] запропоновано

$$I_s(e) = min(I_{s_1}(e), I_{s_2}(e)) \quad (2.20)$$

– якщо відбувається прояв хоча б однієї з властивостей дефекту $s_1, s_2 \in S$ які формують складену властивість *S*, то можна записати

$$I_s(e) = max(I_{s_1}(e), I_{s_2}(e)) \quad (2.21)$$

Модель програмного забезпечення можна описати у вигляді сутностей, їх атрибутів і зв'язків між ними.

За основу пропонується взяти модель [20], що передбачає застосування моделі DDHM шляхом розширення метамоделі Hismo (рис. 2.3). Дана модель дає змогу враховувати історичні дані щодо виявлених та усунутих дефектів програмного забезпечення.

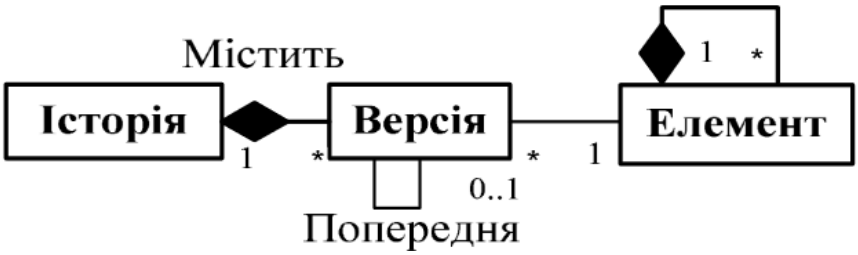

Рис. 2.3. Метамодель Hismo

До складу моделі Hismo, згідно [20], входить:

– елемент конструкції програмного забезпечення на який замінений використовуваний елемент для вивчення його історії;

– версія елемента конструкції програмного забезпечення включає поняття часу до елемента конструкції, відносячи його до історії, версія ідентифікується часовою міткою та містить ідентифікатор історії, частиною якої вона є;

– історія елемента конструкції програмного забезпечення, яка описується множиною його версій, при цьому версія елемента належить тільки одній історії.

Для побудови моделі дефектів програмного забезпечення комп'ютерних систем і виявлення впливу дефектів, що призводять до збоїв програмного забезпечення необхідно розширити модель Hismo шляхом додавання до версій елементів та історії зміни конструктивних елементів програмного забезпечення ще й версій та історій розвитку дефектів. Модель дефектів, в такому випадку, можна представити у вигляді вертикальних та горизонтальних шарів як показано на рис. 2.2.

У вертикальних шарах даної моделі представлено конструктивні елементи програмного забезпечення, їх версії та історії. У горизонтальних шарах розміщено елементи, які для об'єктно-орієнтованих технологій розробки програмного забезпечення можуть представляти пакети класів, елементарні класи, методи. Окрім цього, до горизонтального шару додано шар дефектів, що дає змогу реалізувати трасування їх розвиток за версіями програмного забезпечення.

Формально модель дефектів програмного забезпечення комп'ютерних систем можна визначити як орієнтований мультиграф $MModel = < N, E >$, де $N = \{n_1, n_2, n_3, ..., n_m\}$ – вузли, що представляють собою конструктивні елементи програмного забезпечення; $E = \{e_1, e_2, e_3, ..., e_k\}$ – ребра, що представляють зв'язки між конструктивними елементами.

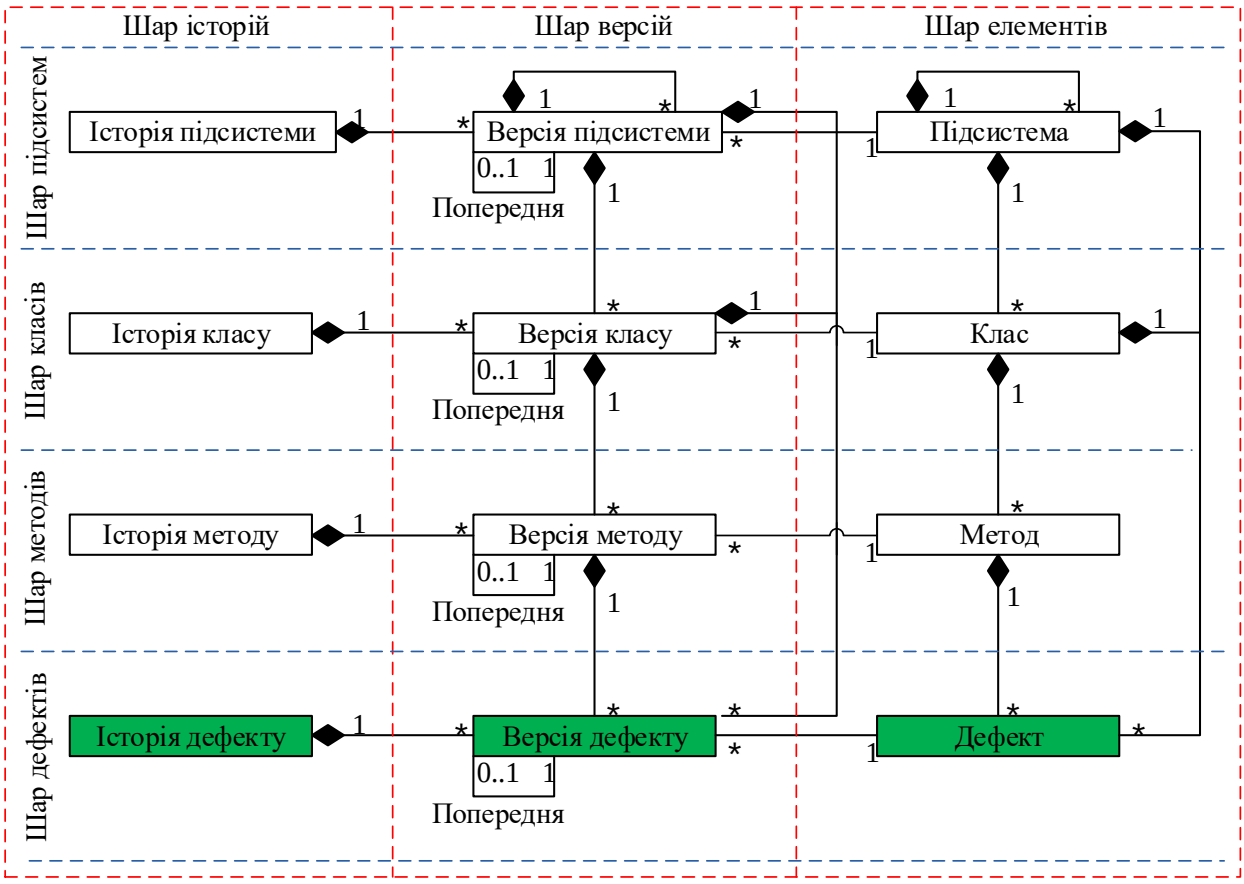

Рис. 2.4. Модель представлення історії дефектів програмного забезпечення комп'ютерних систем

Застосування мультиграфа дає змогу забезпечити побудову зв'язків між вершинами з можливістю поєднання кількох ребер. Для того, що побудувати модель, що враховує історичні аспекти розвитку дефектів застосовується типізація, що вказує на тип елемента конструкції програмного забезпечення, тип дефекту, його історичний розвиток. При визначенні мультиграфа передбачається можливість вкладеності вузлів, що описуються типом ребра «*contains*». Окрім

цього, мультиграф є орієнтованим графом, що дозволяє вказати, що будь-яке ребро має тільки один вузол, а вузол представляє собою мету.

Будь-якому вузлу або ребру можна поставити у відповідність довільну кількість властивостей, що говорить про його навантаженість. Тоді мультиграф, згідно [20], можна представити у вигляді наступних компонентів:

η – сукупність усіх типів вузлів графа, а конкретний екземпляр типу вузла η_i (табл. 2.1);

ρ – сукупність усіх можливих властивостей вузлів, ρ_i – екземпляр можливої властивості вузла;

ε – сукупність всіх типів ребер, ε_i – певний конкрекний екзепляр типу ребра (табл. 2.2).

Виходячи з такої нотації, для об'єктно-орієнтованого програмного забезпечення комп'ютерної системи, вузол *system* буде представляти ціль, а вузол *method* – описує метод класу, що наявний у програмному забезпеченні.

Таблиця 2.1

Типи вузлів моделі

Вид вузла	Опис
system, systemV, systemH	Програмна система, її версія та історія
pack, packV, packH	Пакет, його версія та історія
class, classV, classH	Клас, його версія та історія
method, methodV, methodH	Метод, його версія та історія
attribute, attributeV, attributeH	Атрибут класу, його версія та історія
defect, defectV, defectH	Дефект, його версія та історія

Таблиця 2.2

Типи ребер

Тип ребра	Опис	Можливі пари типів <джерело, мета>
contains	Джерело містить мету	*<system, pack>, <system, class>,* *<pack, pack>,< pack, class>,* *< class, method>, < class, attribute>,* *< pack, defect>,< class, defect>,* *< method, defect>,< packV, packV>,* *< packV, classV >, < classV,* *methodV>,* *< classV, attributeV>,* *< packV, defectV>,< classV, defectV>,* *< methodV,defectV>,< packH, packV>,* *< classH,* *classV>,<methodH,methodV>* *< attributeH, attributeV>,* *< defectH, defectV>,* *< systemH, systemV>,* *< systemV, packV>*
version	Джерело є версією мети	*< packV, pack>, < classV, class>,* *< methodV, method>,* *< attributeV, attribute>* *< defectV,defect>, < systemV, system>*
prev	Джерело є версією, наступною після версії-мети у довільній історії	*< packV, packV>, < classV, classV>,* *< methodV, methodV>,* *< defectV,defectV>, < systemV, system>*
call	Джерело викликає мету	*< method, method>,* *< methodV, methodV>*
inherit	Мета є суперкласом історії	*< class, class>* *< classV, classV>*

Крім прямих зв'язків у мультиграфі передбачено і зворотні зв'язки. Для позначення зворотного зв'язку, коли напрям ребра вказує від цілі до агрегатора нижчого рівня (контейнера), використовується відповідне позначення, наприклад, $\overleftarrow{contains}$. Мультиграф є навантаженим, тому до будь-якого елемента графа можу додаватись довільна кількість властивостей.

Приклади властивостей вузлів, що наведені у [20], наведено у табл. 2.3.

Таблиця 2.3

Властивості вузлів

Тип вузла	Властивість	Опис
systemV, packV	*date*	Дата створення версії
classV, methodV,	*rank*	Номер версії в історії
defectV		
defectV	*dpd*	Ступінь розвитку дефекту (defect progress degree)
	msi	Середня інтенсивність ознак дефекту (mean sign intencity)
	defType	Тип дефекту (defect type)

При виконанні операцій над елементами мультигарфа, згідно [20], можуть застосовуватись функції, які наведено у вигляді табл. 2.4.

Для подальшого використання моделі у [20] визначено множину

$$V = \{n \in N \mid type(n) \in \{packV, classV, methodV\}\},$$

(2.22)

$$DT=\{x|x - \text{тип дефекту}\}$$

Таблиця 2.4

Функції маніпулювання над моделлю

Функція	Опис
$source: E \to N$	Для отримання вузла, джерела ребра
$target: E \to N$	Для отримання вузла, мети ребра
$type: N \to \eta$	Для отримання типу вузла
$type: E \to \varepsilon$	Для отримання типу ребра
$property: N, \rho \to PROPV$	$PROPV$ – множина значень властивостей вузла

Для виявлення зв'язку між вузлами n_1 і n_2, де $n_1, n_2 \in N, \varepsilon_1 \in \varepsilon$ використовується предикат $isEdge: E \times N \times \varepsilon \times N \to Boolean$. Його можна записати наступним чином

$$isEdge(e, n_1, \varepsilon_1, n_2): source(e) = n_1 \wedge target(e) = n_2 \wedge target(e) = \varepsilon_1 \qquad (2.23)$$

При встановленні рівня розвитку дефекту конструкції програмного забезпечення комп'ютерних систем $d \in D$, яким уражений деякий елемент $v \in V$ застосовується функція $Dpd: V \times D \to [0,1000]$, що має наступний вигляд

$$Dpd(v, dt) = \begin{cases} property(dv, dpd), \ якщо \\ (\exists dv \in \{n \in N \mid type(n) = defecV \wedge \\ \wedge\ property(dv, defType = dt\}) \\ (\exists e \in E\ isEdge(e, v, contains, dv); \\ 0, \ в\ інших\ випадках \end{cases} \qquad (2.24)$$

Таким чином, для моніторингу за розвитком дефектів програмного забезпечення обґрунтовано застосування моделі і методу, які наведені у [20], що

дозволяє реалізувати запропоновану модель виявлення впливу дефектів і збоїв програмного забезпечення на надійність комп’ютерних систем.

2.4. Обґрунтування моделей прогнозування дефектів програмного забезпечення комп’ютерних систем

Для прогнозування розвитку дефектів програмного забезпечення, що призводять до збоїв у його роботі, і з можливістю визначення впливу на надійність комп’ютерних систем за моделлю (2.16) можна застосувати наступні моделі:

- Rome Laboratory Model (RLM);
- модель Гефні-Девіса;
- модель Малаї-Дентона;
- модель COCUAL-MO.

Використання моделі RLM щодо прогнозування показників надійності програмного забезпечення здійснює збір та аналіз даних на ранніх етапах розробки, починаючи від ідеї про створення комп’ютерної системи, закінчуючи етапом конструювання (написання коду ПЗ). Модель RLM, згідно [21], представляється щільність дефектів у вигляді

$$D_0 = Z \cdot Y \cdot \prod_{i=1}^{9} D_i \qquad (2.25)$$

Z – тип програмного забезпечення;

Y – середовище реалізації;

D_1 – вимоги до відмовостійкості;

D_2 – вимоги до трасування рішень;

D_3 – вимоги якості;

D_4 – рівень мови програмування;

D_5 – розмір програми;

D_6 – структурованість програмного забезпечення;

D_7 – об’єм повторно-використовуваного коду;

D_8 – складність програмного забезпечення;

D_9 – відповідність стандартам.

Початкова інтенсивність відмов за [21] обчислюється як

$$\lambda_0 = \varphi \cdot K(D_0 \cdot KSLOC) \tag{2.26}$$

φ – середня інтенсивність виконання, поділена на об’єм виконуваного коду;

K – коефіцієнт виявлення дефектів: $1.4 \cdot 10^{-7} \le K \le 10.6 \cdot 10^{-7}$

KSLOC – одиниця розміру ПЗ (тисяча рядків початкового коду).

Особливістю моделі Гефні-Девіса є те, що дані про виконання проекту та інформація про дефекти програмного забезпечення збираються на всіх стадіях життєвого циклу.

Основним критерієм надійності програмного забезпечення при використання моделі Гефні-Дефіса є щільність виявлення дефектів на конкретній стадії життєвого циклу комп’ютерної системи, яка обчислюється за формулою

$$\Delta D_g = E \cdot [exp(-B \cdot (g-1)^2) - exp(-B \cdot g^2)] \tag{2.27}$$

У даному випадку:

g – коефіцієнт виявлення дефектів в залежності від стадії, як приклад, $g = 1$ – стадія формування вимог, $g = 2$ – стадія проектування архітектури, $g = 3$ –стадія конструювання програмного забезпечення, $g = 4$ – модульнк тестування, $g = 5$ – системне тестування.

Значення *B* є оберненим значенням до подвійного квадрату τ_p – показника моменту часу виявлення дефекту, який обчислюється як

$$B = 1/2\tau_p^2 \tag{2.28}$$

Показник τ_p відповідає піку на кривій Релея (рис. 2.5), що передбачає виявлення 39% дефектів програмного забезпечення комп'ютерної системи.

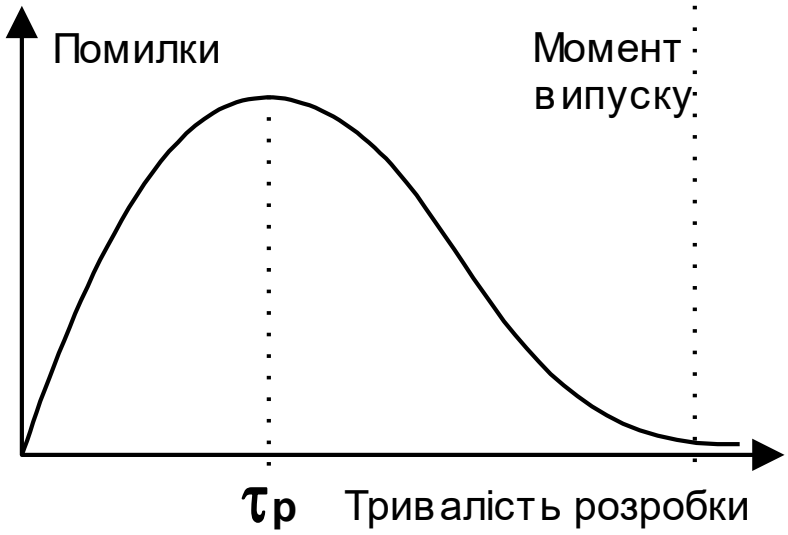

Рис. 2.5. Крива Релея

Основними факторами прогнозування надійності програмного забезпечення комп'ютерних систем при застосуванні моделі Гефні-Девіса є:

- *E* – інтенсивність внесення дефектів;
- стадія ЖЦ;
- розмір ПЗ (у *SLOC*);
- інтенсивність виявлення дефектів за кривою Релея.

Для прогнозування кількості не виявлених дефектів на деякій стадії *g* можна використати формулу

$$D_g = E \cdot exp(-B \cdot g^2) \cdot SLOC_g] \tag{2.29}$$

$SLOC_g$ – кількість стрічок програмного коду, визначена на стадії g.

Модель Малаї-Дентона може ефективно застосовуватись при прогнозуванні дефектів програмного забезпечення комп'ютерних систем починаючи зі стадії кодування, закінчуючи випуском готової комп'ютерної системи. В основі моделі лежать принципи визначення модифікаторів щільності дефектів програмного забезпечення. Для обчислення щільності дефектів програмного забезпечення використовується формула

$$D_0 = C \cdot \prod_{i=1}^{5} D_i, \tag{2.30}$$

де C – константа пропорційності кількості коду і кількості дефектів (кількість дефектів у KSLOC), $6 \le C \le 20$;

D_1 – етап тестування програмного забезпечення комп'ютерної системи;

D_2 – зрілість команди розробників (кваліфікація, досвід);

D_3 – зрілість процесу розробки за CMM;

D_4 – структура програмного забезпечення комп'ютерної системи (мова, складність, модульність та ін.)

D_5 – змінюваність вимог.

У випадку використання низькорівневих мов програмування, наприклад Assembler, передбачається, що код програмного компоненту містить на 40% більше дефектів.

Модель COCUAL-MO є розвитком та розширенням моделі COCOMO II і дає змогу прогнозувати надійність програмного забезпечення, починаючи від зародження концепції і до введення програмної складової комп'ютерної системи в експлуатацію. При застосуванні даної моделі основними критеріями прогнозування надійності програмного забезпечення виступають кількість нетривіальних дефектів за результатами виконання стадій виявлення та аналізу вимог, проектування архітектури та кодування.

Модель COCUAL-MO містить дві частини:

– модель внесення дефектів;

– модель виявлення дефектів.

Основними параметрами моделі COCUAL-MO, згідно [21], є:

I_s – розмір програмного забезпечення, що може бути обчислений у *KSLOC* або *УОФ;*

A_j – базове значення щільності дефектів для артефакту j ($j=1$ – для вимог, $j=2$ – для проекту, $j=3$ – для коду)

D_{ij} – модифікатор щільності дефектів, пов'язаних з чинником i ($i=1, 2,...,21$) для артефакту j ($j = 1,2,3$) за категоріями – платформа, продукт, персонал, проект.

M_{ij} – частка дефектів артефакту j, яка видаляється завдяки застосуванню методології i ($i=1,2,3$).

У моделі COCUAL-MO оцінку загальної кількості внесених дефектів обчислюють за формулою

$$N = \sum_{j=1}^{3} A_j \cdot I_s^{B_j} \cdot \prod_{i=1}^{21} D_{ij}, \tag{2.31}$$

$B_j = 1$ – у поточній версії моделі.

Для встановлення значення оцінки кількості внесених дефектів у артефакті j, N_j використовується формула

$$N_j = A_j \cdot I_s^{B_j} \cdot \prod_{i=1}^{21} D_{ij} \tag{2.32}$$

З метою врахування впливу кожного чинника i на щільність дефектів у артефакті j можливе застосування шкали значень від *VH* – дуже високий до *VL* – дуже низький. Згідно [21], значення, більші від 1, свідчать про зростання

щільності дефектів внаслідок впливу фактору, значення, менші за 1 – про його зменшення, рівне 1 – номінальне значення.

Рівень ефективності застосування методології *i* щодо виявлення дефектів артефакту *j* можу бути визначений нечіткими значеннями *VL* – дуже низька ефективність, *EH* – екстра висока ефективність. Значення ефективності методологій для чисельного вираження лежать в інтервалі [0,1), де 0 – найнижча ефективність.

Для кількісного вираження загальної кількості дефектів, N_j, які не були усунутими для артефакту *j* визначається за формулою:

$$N_j = C_j \cdot I_{sj} \cdot \prod_i (1 - U_{ij}) \tag{2.33}$$

C_j – базове значення залишкової кількості дефектів *j*-го артефакту.

U_{ij} – частка усунутих дефектів *j*-го артефакту за допомогою методології *i*.

Переваги, розглянутих вище моделей прогнозування надійності програмного забезпечення, зокрема показників інтенсивності та щільності дефектів, які можуть призводити до збоїв програмного забезпечення і впливати на надійність комп'ютерної системи, в порівнянні з регресійними моделями, полягають в наступному:

– відсутність від'ємних або нульових значень при прогнозуванні показників кількості, щільності та інтенсивності дефектів програмного забезпечення комп'ютерних систем;

– можливість одночасного врахування показників надійності програмного забезпечення;

– можливість застосування моделей у випадку неповної інформації та в умовах невизначеності, що враховує значення невідомого показника рівним 1, як середньої оцінки впливу дефекту;

– можливість одержання кількісних значень оцінок на основі аналізу великої сукупності експериментальних даних.

До недоліків розглянутих моделей прогнозування дефектів програмного забезпечення комп’ютерних систем належать:

– залежність і придатність до застосування кількісних характеристик надійності програмного забезпечення лише у випадку уже виконаних проектів;

– необхідність налаштування параметрів моделей на основі історичних даних попередніх проектів конкретної фірми-розробника комп’ютерних систем, зокрема програмної складової

– неадаптованість моделей прогнозування дефектів до сучасних моделей життєвого циклу розробки програмного забезпечення комп’ютерних систем;

– необхідність впровадження процесу вимірювання на усіх етапах життєвого циклу, що вимагає високої зрілості організації процесів на кожному з етапів життєвого циклу.

2.5. Висновки до розділу

Основні наукові і практичні результати, які одержано у даному розділі полягають в наступному:

1. Визначено та формалізовано показники надійності програмного забезпечення комп’ютерних систем, що дають змогу кількісно виражати наявність, щільність та інтенсивність дефектів, які можуть призводити до збоїв у роботі програмних складових та негативно впливати на надійність комп’ютерної системи в цілому.

2. Побудовано та обґрунтовано модель впливу дефектів програмного забезпечення на надійність комп’ютерної системи, що забезпечує можливість встановлення шляху поширення дефекту програмного забезпечення на інші компоненти комп’ютерної системи і дає змогу прогнозувати ймовірність виникнення негативного впливу на надійність комп’ютерної системи.

3. Обґрунтовано модель і метод моніторингу дефектів програмного забезпечення комп’ютерних систем на основі модифікованої моделі Hismo, що дає змогу аналізувати розвиток і вплив дефекту програмного забезпечення на

надійність комп’ютерної системи у часі та з врахуванням версій програмного забезпечення.

4. Обґрунтовано моделі прогнозування дефектів програмного забезпечення комп’ютерних систем, що дають змогу на «ранніх» і «пізніх» стадіях розробки комп’ютерних систем виявляти можливі загрози збою програмного забезпечення та знижувати їх негативний вплив на надійність комп’ютерної системи.

РОЗДІЛ 3
ЗАСІБ ПІДТРИМКИ ПРОЦЕСУ ВИЗНАЧЕННЯ ВПЛИВУ ЗБОЇВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА НАДІЙНІСТЬ КОМП'ЮТЕРНИХ СИСТЕМ

3.1. Аналіз вимог до засобу підтримки процесу визначення впливу збоїв програмного забезпечення на надійність комп'ютерних систем

При визначенні вимог до програмного засобу підтримки процесу визначення впливу збоїв програмного забезпечення на надійність комп'ютерних систем необхідно визначити основні сутності предметної області, описати їх взаємозв'язок та змоделювати відношення між учасниками процесу та функціональними можливостями системи.

Для представлення функціональних вимог до програмного засобу запропоновано скористатись нотаціями мови UML, зокрема use case діаграмами.

У загальному випадку, програмний засіб підтримки процесу підтримки процесу визначення впливу збоїв програмного забезпечення на надійність комп'ютерних систем передбачає формування довідників, необхідних для маніпулювання описом дефектів, представлення моделей дефектів та ін.

На рис. 3.1 представлено функціональні можливості розробників комп'ютерної системи щодо керування вхідними даними процесу визначення дефектів та подальшого аналізу їх впливу на комп'ютерну систему.

Розробники комп'ютерних систем, або особи, що відповідають за забезпечення надійності програмного забезпечення комп'ютерних систем повинні мати можливість сформувати набори метрик для оцінювання дефектів, зокрема щодо щільності та інтенсивності дефектів. Для прогнозування виникнення можливих дефектів необхідно забезпечити створення параметрів для їх опису. Важливим також є інформація про архітектурні компоненти комп'ютерних систем, зокрема програмне забезпечення, апаратні пристрої та канали передачі даних.

Рис. 3.1. Вимоги до підсистеми довідників

Розробники повинні мати можливість маніпулювати зміною параметрів моделей прогнозування дефектів програмного забезпечення, вносити нові моделі, змінювати архітектурні компоненти системи, а також формувати набори відомих дефектів програмного забезпечення.

При формуванні залежностей і можливості визначення потенційного поширення дефекту програмного забезпечення у комп’ютерній системі необхідно забезпечити побудову відношень, які представлені вимогами на рис. 3.2.

Перед встановлення відношень між компонентами комп’ютерної системи, повинні бути сформованими довідники, які описують характеристики цих компонентів. Важливим при формуванні зв’язків між компонентами комп’ютерної системи є забезпечення можливості їх зміни.

У результаті реалізації наведених на рис. 3.2 вимог буде забезпечена можливість встановлення впливу збоїв програмного забезпечення на апаратне забезпечення, або навпаки. Крім, того можливе виявлення збоїв і на рівні каналів

передачі даних, у випадку переконаності у тому, що програмне і апаратне забезпечення функціонують в нормальному режимі.

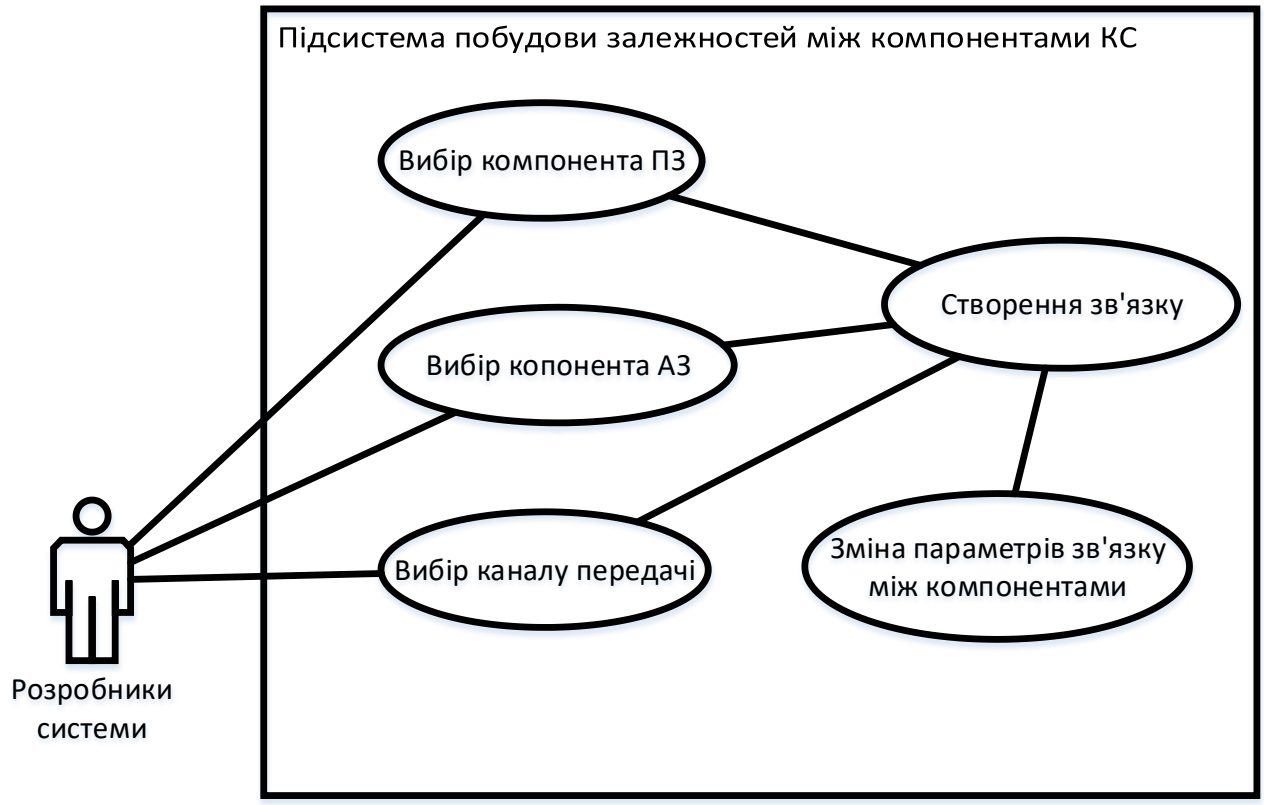

Рис. 3.2. Вимоги до підсистеми формування залежностей між компонентами комп’ютерної системи

Важливим, з точки зору забезпечення надійної роботи комп’ютерної системи, є обчислення та прогнозування імовірних значень дефектів програмного забезпечення, що призводять до збоїв у його роботі.

Основні функціональні можливості, які при цьому необхідно забезпечити, полягають у наступному:

- можливість вказати версію комп’ютерної системи і її складових;
- можливість обрати показники надійності;
- здатність обрати існуючі дефекти, або типові із довідників;
- можливість обрати метрики оцінювання;
- провести безпосереднє обчислення оцінок дефектів;

– сформувати результати оцінювання дефектів і їх вплив на надійність комп’ютерної системи.

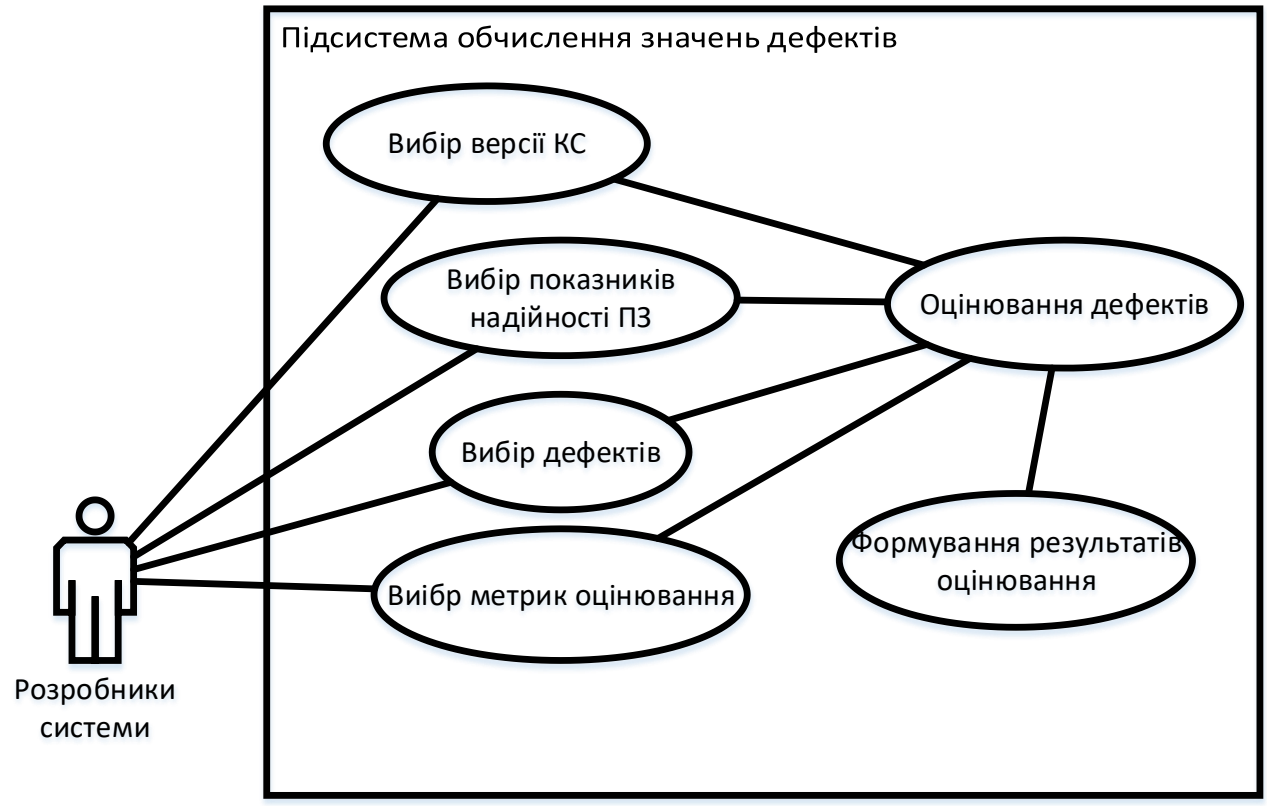

Рис. 3.3. Вимоги до обчислення значень дефектів

На основі представлених на рис. 3.1, рис. 3.2 і рис. 3.3 функціональних вимог необхідно провести додатковий аналіз і спроектувати базу даних для забезпечення можливості зберігання та маніпулювання даними.

3.2. Проектування бази даних підтримки процесу оцінювання впливу дефектів програмного забезпечення на надійність комп’ютерних систем

Для проектування бази даних підтримки процесу оцінювання впливу дефектів програмного забезпечення на надійність комп’ютерних систем пропонується скористатись підходом реляційних баз даних.

При цьому таблиці будуть відображати сутності предметної області, стовпці у таблицях – властивості сутності.

При проведенні аналізу предметної області та процесів оцінювання впливу дефектів програмного забезпечення на надійність комп'ютерних систем визначено 8 сутностей.

Для представлення сутності «Комп'ютерна система» використовується опис, який наведено у табл. 3.1.

Таблиця 3.1

«Комп'ютерна система» (Computer System)

Атрибут	Тип	Примітка
ID_CS	int	PK
CSName	varchar(max)	
Version	varchar(20)	
UpdatingDate	date	

У табл. 3.1 поле ID_CS – первинний ключ, CSName – назва комп'ютерної системи, Version – версія комп'ютерної системи, UpdatingDate – дата випуску версії комп'ютерної системи.

Для представлення складових комп'ютерної системи, зокрема програмного забезпечення, апаратних пристроїв та каналів зв'язку визначено відповідні таблиці. У табл. 3.2 наведено структуру для збереження даних про програмні складові комп'ютерної системи.

Таблиця 3.2

«Програмні складові КС» (Software)

Атрибут	Тип	Примітка
ID_SW	int	PK
ID_CS	int	FK
ComponentName	varchar(100)	

Продовж. табл. 3.2

Атрибут	Тип	Примітка
Type	varchar(25)	
InputData	varchar(max)	
OutputData	varchar(max)	
ComponentVersion	varchar(25)	
UpdatingDate	date	

У даному випадку ID_SW – ідентифікатор складової програмного забезпечення, ID_CS – зовнішній ключ, що посилається на ідентифікатор таблиці «Комп'ютерна система», ComponentName – назва програмного компоненту, Type – тип програмної складової, InputData – вхідні параметри програмного компоненту, OutputData – вихідні параметри програмного компоненту, ComponentVersion – версія програмного компоненту, UpdatingDate – дата створення версії програмного компоненту.

У табл. 3.3 представлено структуру таблиці для збереження даних про апаратні складові комп'ютерної системи.

Таблиця 3.3

«Апаратні складові КС» (Hardware)

Атрибут	Тип	Примітка
ID_HW	int	PK
ID_CS	int	FK
HWName	varchar(100)	
Type	varchar(25)	
InputData	varchar(max)	
OutputData	varchar(max)	
UpdatingDate	date	

Первинним ключем у табл. 3.3 є атрибут ID_HW, зовнішнім – ID_CS, HWName – назва апаратного пристрою, Type – тип апаратного пристрою, InputData – опис вхідних даних, OutputData – опис вихідних даних, UpdatingDate – дата впровадження апаратного пристрою у версію комп'ютерної системи.

У табл. 3.4 наведено структуру таблиці, що описує канали комунікації всередині комп'ютерної системи.

Таблиця 3.4

«Канали зв'язку»(Channel)

Атрибут	Тип	Примітка
ID_Ch	int	PK
ID_CS	int	FK
ChName	varchar(100)	
Type	varchar(25)	
Version	varchar(25)	
Protocol	varchar(30)	
Description	varchar(max)	

У табл. 3.4 первинним ключем є атрибут ID_Ch, а зовнішнім – ID_CS. Поле ChName містить назву каналу зв'язку, Type – тип каналу зв'язку, Version – версія протоколу передачі даних, Protocol – назва протоколу передачі даних, Description – додатковий опис можливостей каналу передачі даних.

Для зберігання та маніпулювання метриками щодо оцінювання дефектів програмного забезпечення та надійності комп'ютерних систем, створено таблицю «Метрики», структуру якої наведено у табл. 3.5. Основними складовими таблиці є поля:

- ідентифікатор метрики – ID_M;
- назва метрики – MetricName;
- формула для обчислення значення метрики – Formula;
- опис метрики – Description.

Таблиця 3.5

«Метрика» (Metrics)

Атрибут	Тип	Примітка
ID_M	int	PK
MetricName	varchar(100)	
Formula	varchar(100)	
Description	varchar(max)	

Для зберігання даних про наявні або прогнозовані дефекти програмного забезпечення комп'ютерної системи створено таблицю «Дефект» (Defect), структуру якої показано у табл. 3.6.

Таблиця 3.6

«Дефект» (Defect)

Атрибут	Тип	Примітка
ID_Def	int	PK
DefectName	varchar(100)	
DefectNum	int	
ID_SW	int	FK
ID_HW	int	FK
ID_Ch	int	FK
ID_M	int	FK
Value	float	
Description	varchar(MAX)	

У табл. 3.6 ідентифікатором виступає атрибут ID_Def, зовнішні ключі ID_SW, ID_HW, ID_Ch, ID_M є посиланнями на первинні ключі відповідно таблиць: «Програмні складові КС», «Апаратні складові КС», «Канали зв'язку» та «Метрики». Атрибут DefectName містить значення назв дефектів, а DefectNum – номер дефекту у комп'ютерній системі. Атрибут Value – зберігає значення, що

представляє рівень розвитку дефекту програмного забезпечення комп'ютерної системи, а Description – містить додаткову інформацію про дефект.

Таблиця «Оцінка дефекту» (Evaluation) містить інформацію про кількісні характеристики дефекту програмного забезпечення комп'ютерної системи і має структуру, як показано у табл. 3.7.

Таблиця 3.7

«Оцінка дефекту» (Evaluation)

Атрибут	Тип	Примітка
ID_Ev	int	PK
ID_Def	int	FK
Type	varchar(35)	
Mark	float	
Acceptability	bit	
Max_Value	varchar(50)	
Min_Value	varchar(50)	
ID_RC	Int	FK
Description	varchar(MAX)	

Зміст атрибутів таблиці «Оцінка дефекту» наступний:

– ID_Ev – ідентифікатор оцінки дефекту;

– ID_Def – зовнішній ключ, що визначає дефект;

– Type – тип оцінки дефекту;

– Mark – значення оцінки дефекту;

– Acceptability – прийнятність оцінки, що виражає ступінь розвитку дефекту;

– Max_Value – верхній поріг для оцінки дефекту;

– Min_Value – нижній допустимий поріг для оцінки дефекту;

– ID_RC – зовнішній ключ, що визначає критерій надійності;

– Description – додаткова інформація про оцінку дефекту;

Остання таблиця бази даних описує властивості критеріїв надійності програмного забезпечення та комп'ютерної системи в цілому. У табл. 3.8 наведено структуру таблиці «Критерії надійності».

Таблиця 3.8

«Критерії надійності»(ReliabilityCriteria)

Атрибут	Тип	Примітка
ID_RC	int	PK
CriteriaName	varchar(35)	
Formula	varchar (100)	
Description	varchar(MAX)	

У табл. 3.8 ID_RC – первинний ключ, CriteriaName – назва критерію надійності, Formula – функція для обчислення значення критерію надійності, Description – інформація щодо застосування критерію надійності.

ER-діаграма бази даних наведена на рис. 3.4.

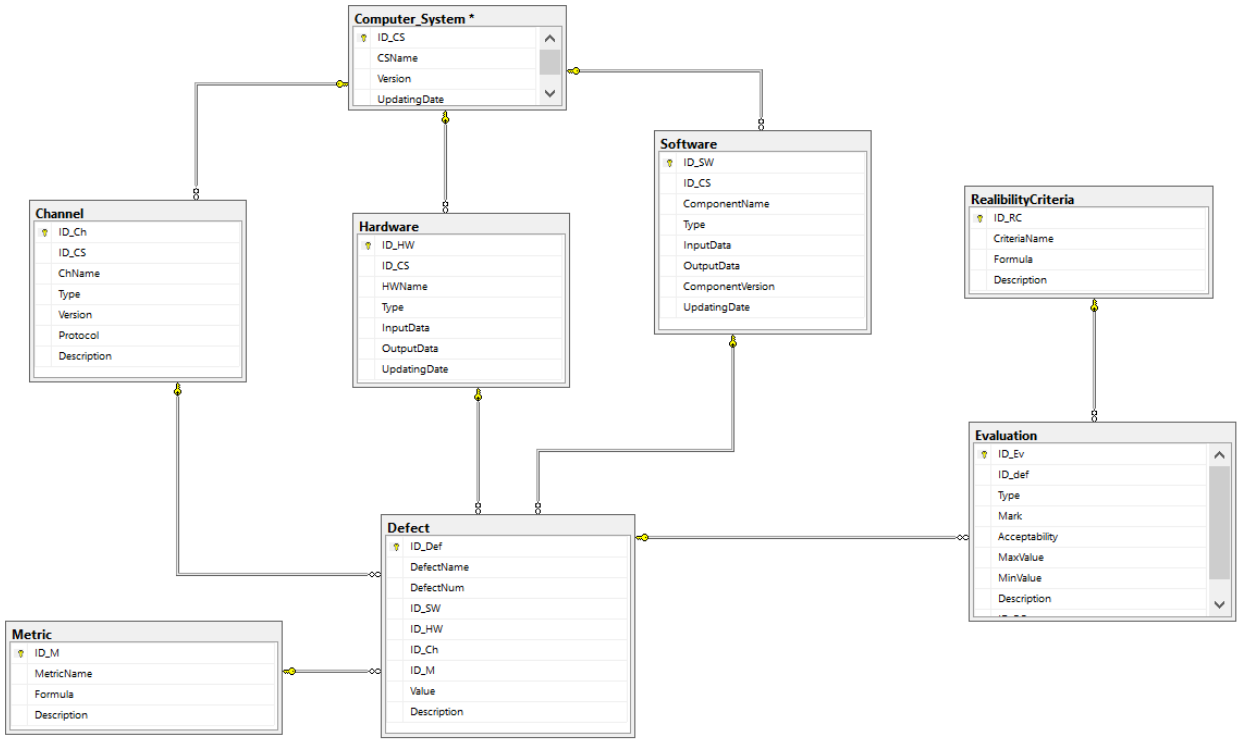

Рис. 3.4. ER-діаграма бази даних

Схему бази даних нормалізовано та приведено до третьої нормальної форми, що забезпечує цілісність та не значну надлишковість даних. Наступний етап полягає у проектуванні архітектури програмного засобу підтримки процесу визначення впливу дефектів програмного забезпечення на надійність комп’ютерної системи.

3.3. Проектування архітектури засобу підтримки процесу визначення впливу дефектів програмного забезпечення на надійність комп’ютерної системи

При побудові архітектури програмного засобу визначення впливу дефектів програмного забезпечення на надійність комп’ютерних систем пропонується скористатись шарами Фаулера.

Враховуючи функціональні вимоги та структуру бази даних архітектуру засобу спроектовано так, як показано на рис. 3.5.

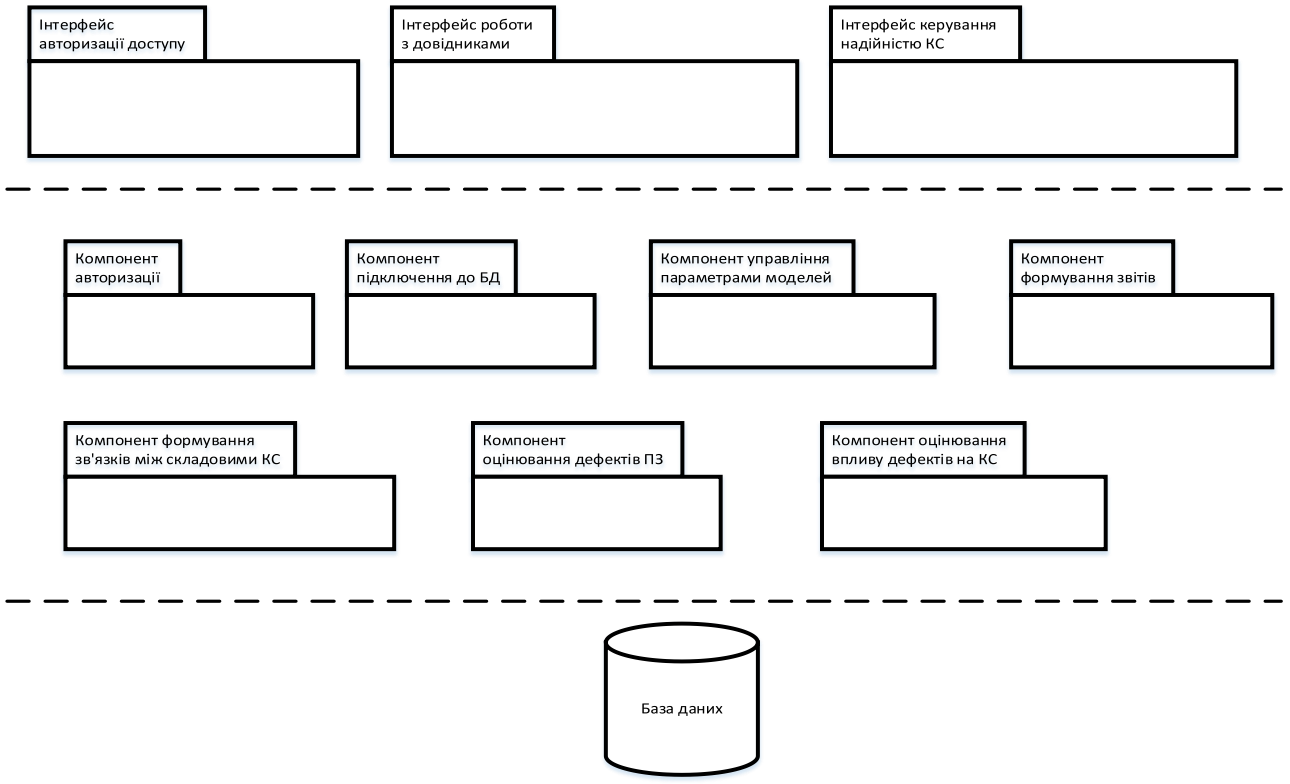

Рис. 3.5. Архітектура засобу за шарами Фаулера

За концепцією проектування архітектури програмного забезпечення за шарами Фаулера, на найвищому рівні (рівень представлення) наведено інтерфейси користувачів, зокрема:

– інтерфейс авторизації у системі;
– інтерфейс роботи з довідниками;
– інтерфейс керування надійністю комп’ютерної системи.

Інтерфейс авторизації передбачає використання елементів вводу даних логіна і пароля для того, щоб одержати доступ до процесів управління процесом визначення впливу дефектів програмного забезпечення на надійність комп’ютерної системи.

Інтерфейс роботи з довідниками забезпечує доступ до форм для заповнення первинних даних щодо опису дефектів, компонентів комп’ютерної системи та оцінювання впливу дефекту програмного забезпечення на надійність комп’ютерної системи.

Інтерфейс керування надійністю комп’ютерної системи передбачає наявність елементів керування щодо безпосереднього оцінювання критеріїв надійності та обчислення кількісних значень щодо прогнозованої та реальної кількості виявлених дефектів.

Середній рівень шарів Фаулера відповідає за логіку функціонування програмного засобу. Компонент авторизації виконує функцію перевірки введених ідентифікаційних даних користувача програмного засобу.

Компонент підключення до бази даних забезпечує перевірку з’єднання з базою даних і доступністю до необхідних даних при використанні програмного засобу підтримки процесу визначення впливу збоїв програмного забезпечення на надійність комп’ютерних систем.

Компонент управління моделями надійності забезпечує можливість формування параметрів моделей, додавання нових моделей до існуючих та здатність порівняння результатів прогнозування дефектів за різними моделями.

Компонент формування зв’язків між компонентами комп’ютерних систем дають змогу встановити відношення між програмним та апаратним забезпеченням через відповідні канали передачі даних.

Компонент оцінювання дефектів програмного забезпечення призначений для визначення критеріїв надійності програмного забезпечення шляхом обчислення кількості дефектів, їх типів та потенційного розвитку.

Компонент оцінювання впливу дефектів на комп’ютерну систему виконує функції щодо виявлення шляхів поширення дефектів програмного забезпечення, які призводять до його збоїв, на інші складові комп’ютерної системи.

Компонент формування звітів щодо впливу дефектів програмного забезпечення на надійність комп’ютерної системи формує кількісні показники надійності за компонентами комп’ютерної системи.

На найнижчому рівні шарів Фаулера знаходиться база даних, призначення та схему якої розглянуто у п. 3.2.

3.4. Розробка користувацьких інтерфейсів програмного засобу

Для того, щоб почати використовувати спроектований програмний засіб визначення впливу збоїв програмного забезпечення на надійність комп’ютерної системи необхідно пройти авторизацію. Вікно авторизації показано на рис. 3.6.

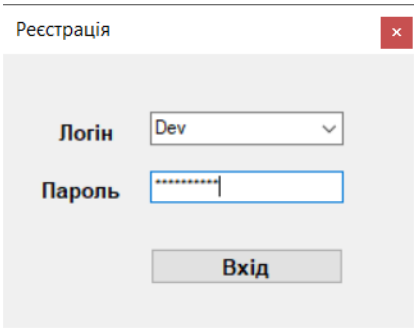

Рис. 3.6. Вікно авторизації користувача

Після успішної авторизації, у користувача з’являється можливість використовувати функції, які передбачено use case діаграмами та архітектурою

програмного засобу. На рис. 3.7 наведено основну форму для роботи із засобом визначення впливу збоїв і дефектів програмного забезпечення на надійність комп'ютерних систем.

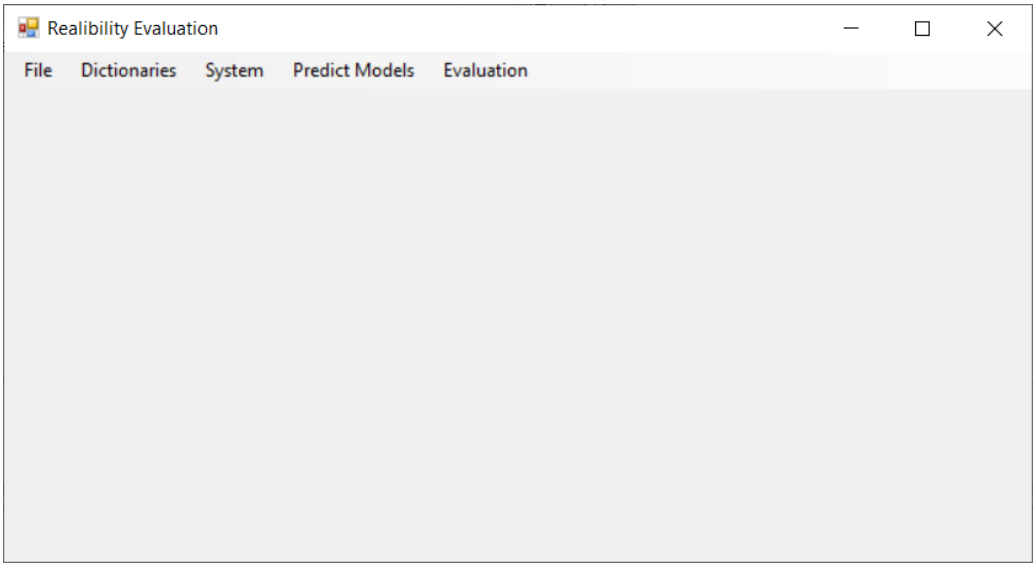

Рис. 3.7. Головна форма програмного засобу

У меню «File» (рис. 3.8) можна встановити з'єднання з базою даних або вийти з програмного засобу. У випадку коректного з'єднання з БД буде виведено повідомлення, як показано на рис. 3.9 та рис. 3.10.

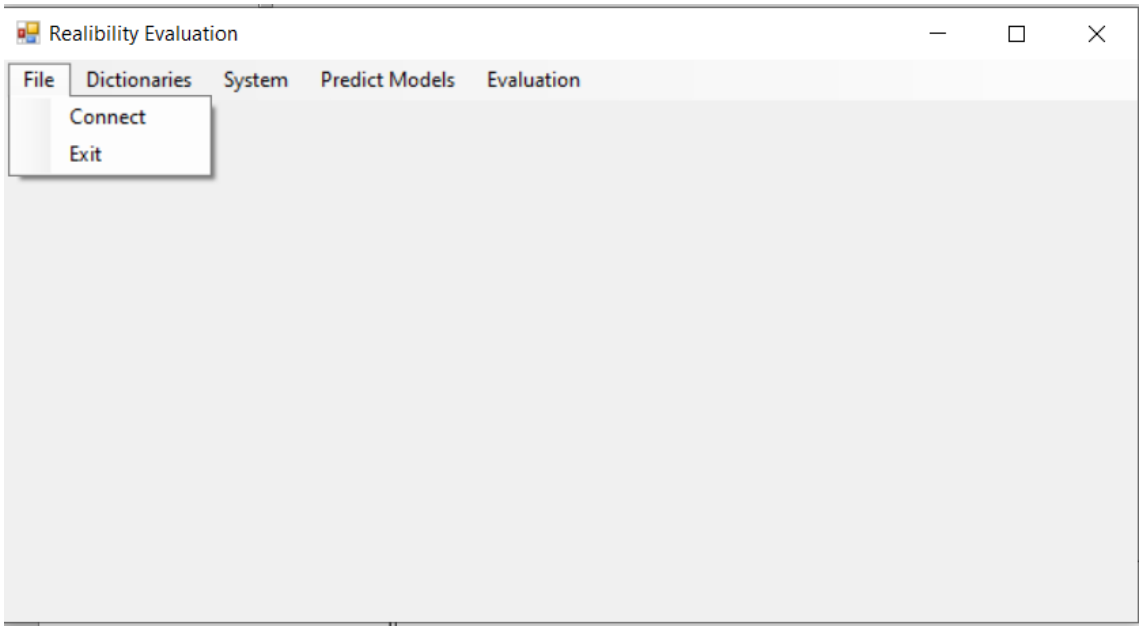

Рис. 3.8. Меню «File»

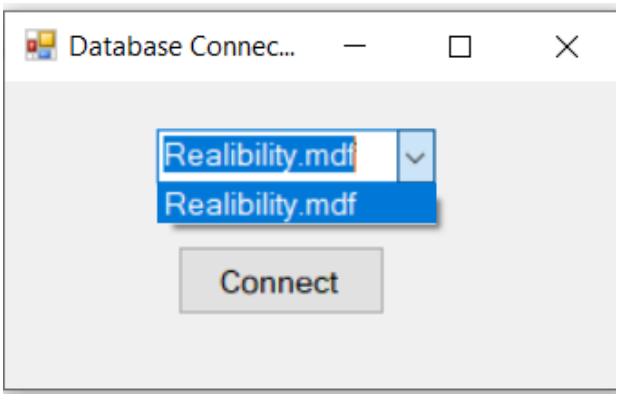

Рис. 3.9. Вікно з’єднання з БД

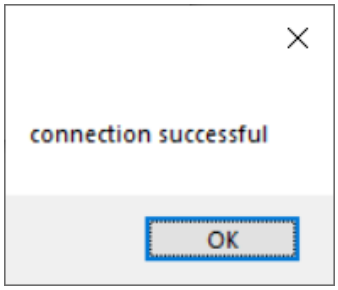

Рис. 3.10. Повідомлення успішного з’єднання

Меню «Dictionary» містить довідники для вхідних даних щодо обчислення інтенсивності та щільності дефектів програмного забезпечення комп’ютерних систем.

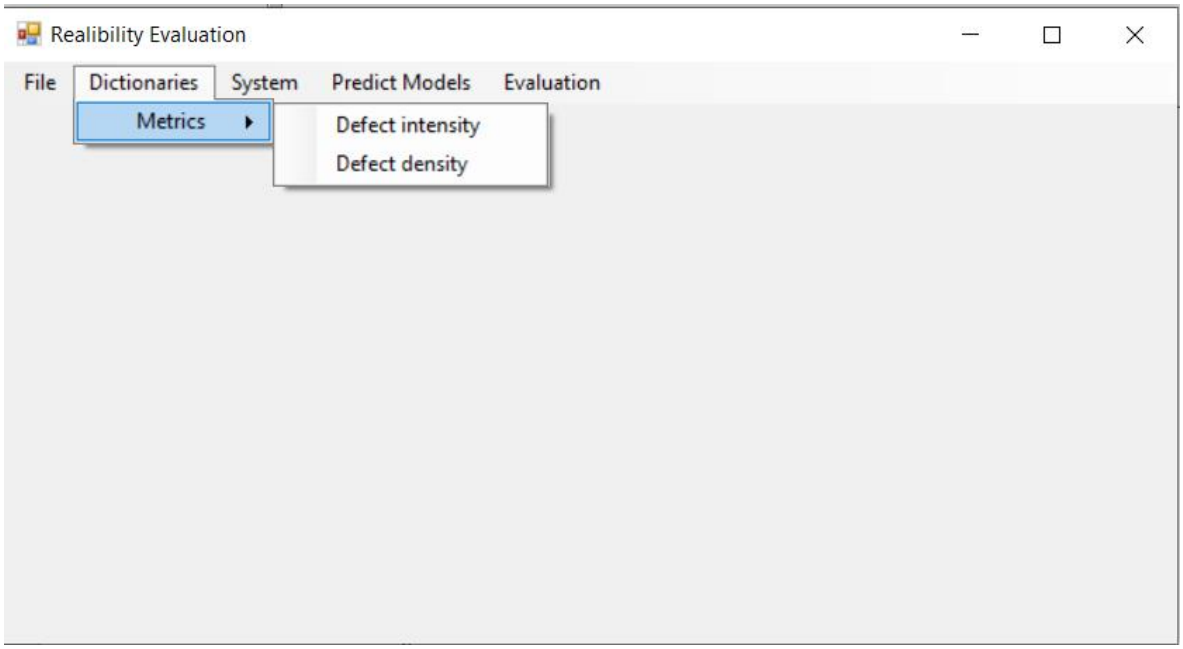

Рис. 3.11. Меню «Dictionaries»

Структура меню «System» представлена на рис. 3.12 і містить довідники щодо компонентів комп’ютерної системи: програмне забезпечення, апаратне забезпечення, канали зв’язку. Крім того, дана форма забезпечує можливість побудувати зв’язки між компонентами комп’ютерної системи та дефектами.

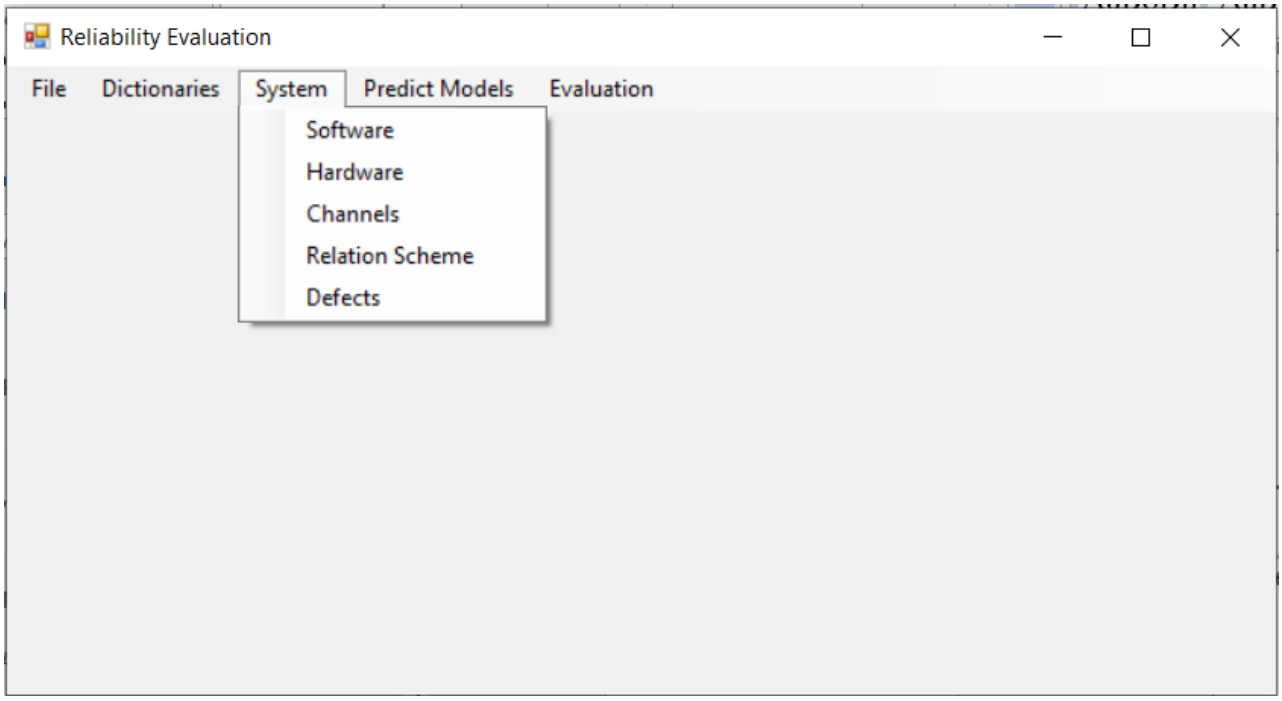

Рис. 3.12. Меню «System»

Меню «Predict Models» (рис. 3.13) містить інформацію про моделі прогнозування надійності програмного забезпечення, зокрема:

- Rome Laboratory Model (RLM);
- модель Гефні-Девіса;
- модель Малаї-Дентона;
- модель COCUAL-MO.

Дане меню дозволює сформувати параметри даних моделей, обчислити відповідні показники щільності та інтенсивності дефектів програмного забезпечення.

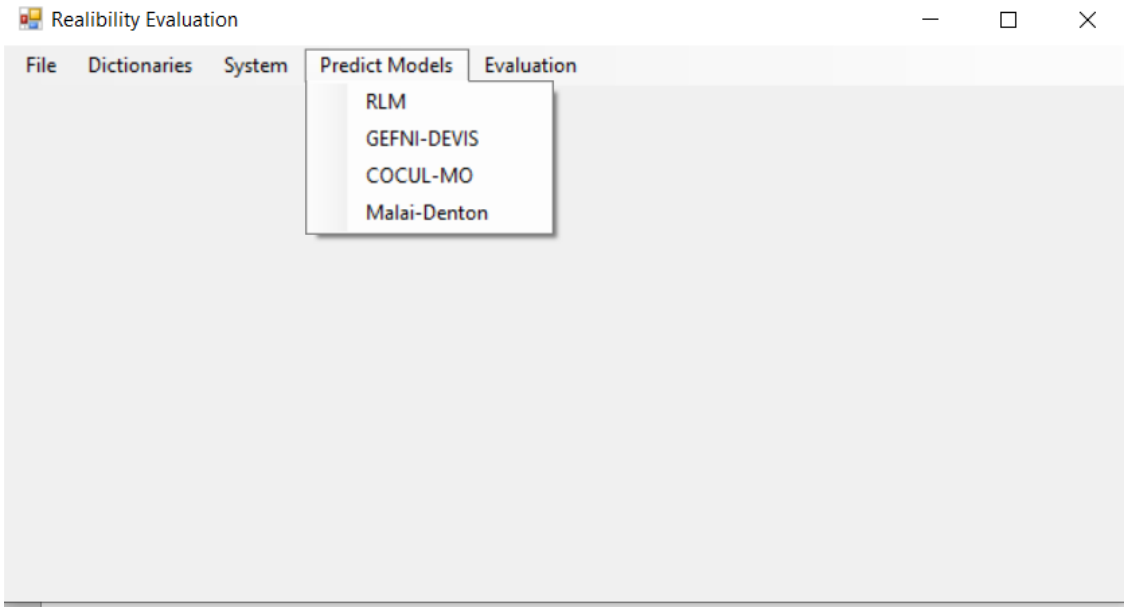

Рис. 3.13. Меню «Predict Models»

Приклад формування характеристик компонентів програмного забезпечення комп'ютерної системи представлено на рис. 3.14.

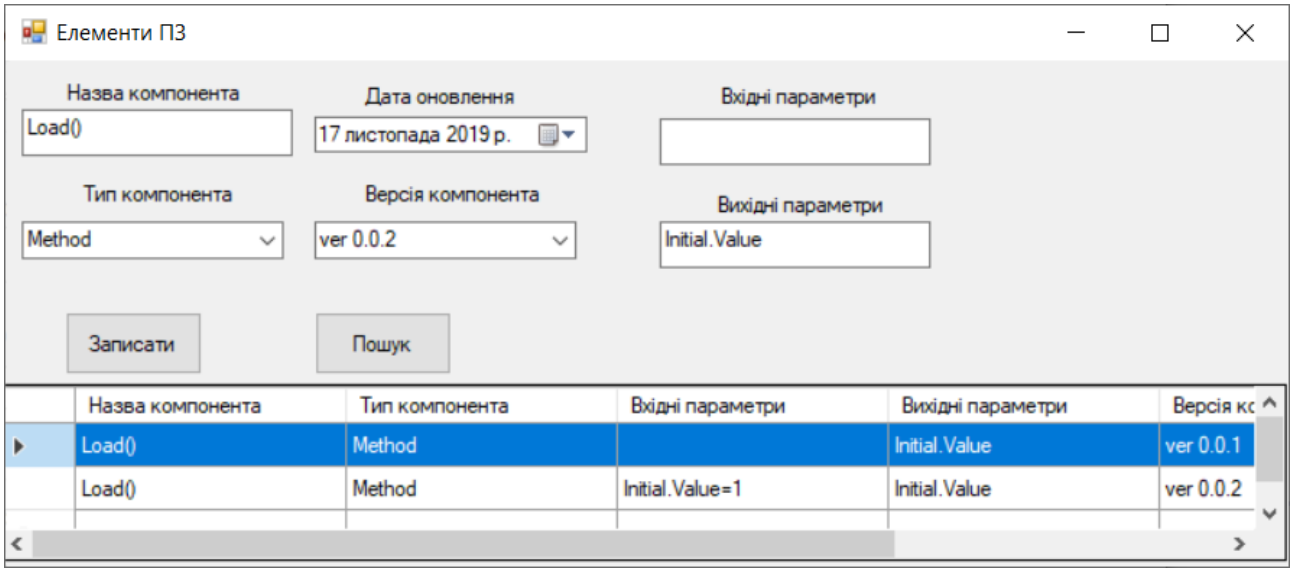

Рис. 3.14. Форма опису компонента програмного забезпечення

Приклад формування метрики для оцінювання дефектів або надійності програмного компоненту комп'ютерної системи наведено на рис. 3.15.

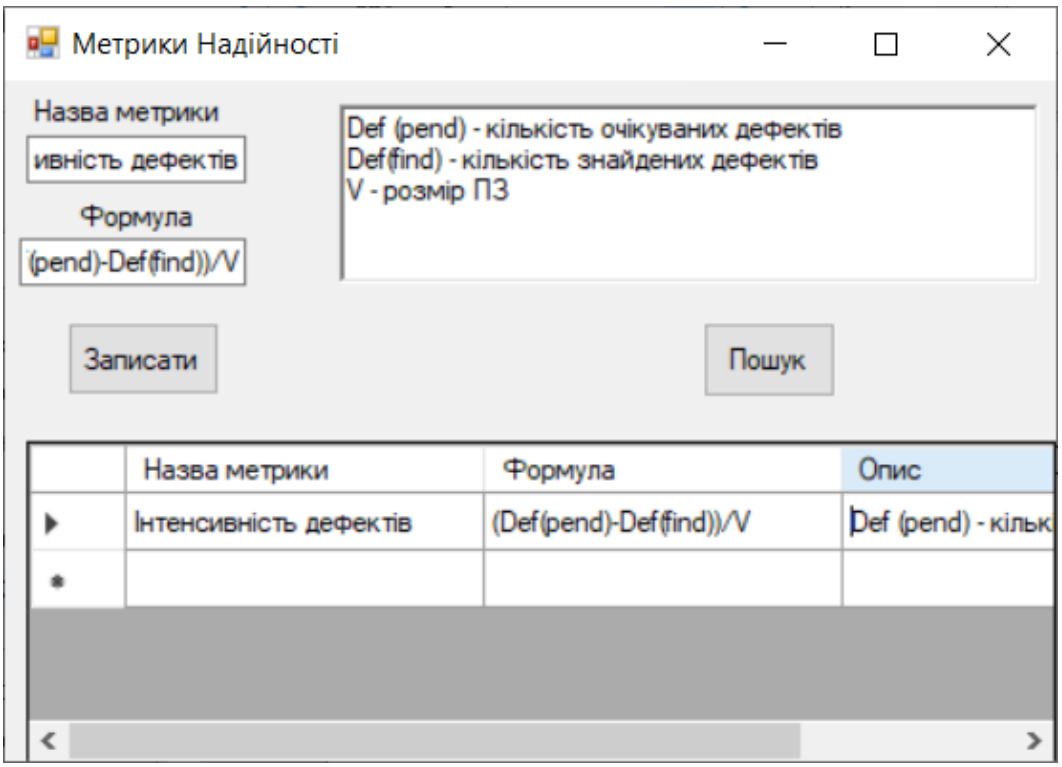

Рис. 3.15. Форма створення метрик надійності

На рис. 3.16 представлено опис дефекту програмного забезпечення, що описує помилку ініціалізації камери у комп'ютерній системі.

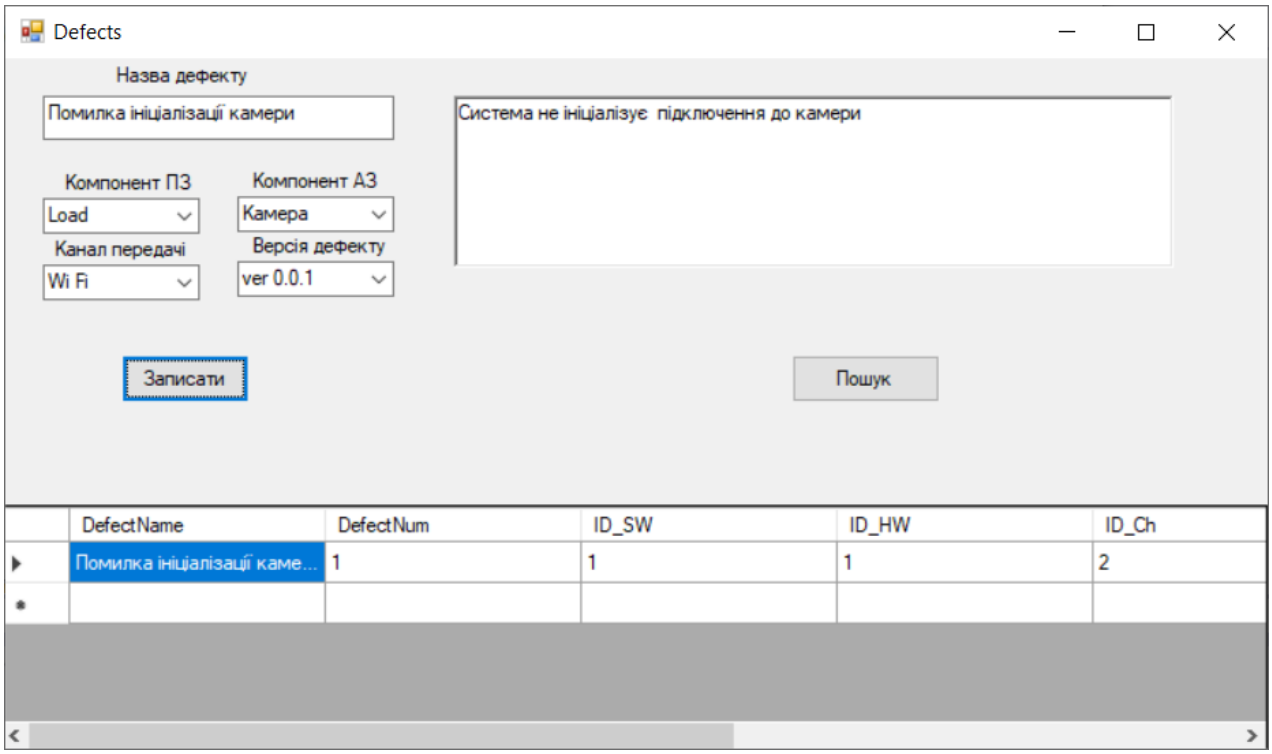

Рис. 3.16. Опис дефекту програмного забезпечення комп'ютерної системи

Таким чином, розроблено програмний засіб, що дає змогу визначити вплив дефектів програмного забезпечення, що призводять до їх збоїв на надійність комп'ютерних систем.

3.5. Експериментальні дані

Розглянемо приклад застосування розроблених моделей, методів і засобу для комп'ютерної системи з наступними характеристиками програмного забезпечення, які наведені у табл. 3.9.

Таблиця 3.9

Характеристики програмного забезпечення комп'ютерної системи

Характеристика	Значення
Розмір програмного забезпечення	136,000 р. коду
Кількість методів/інтерфейсів	14,221
Кількість класів	2,522
Кількість компонентів	143

Розвиток дефектів програмного забезпечення комп'ютерної системи показано на рис. 3.17.

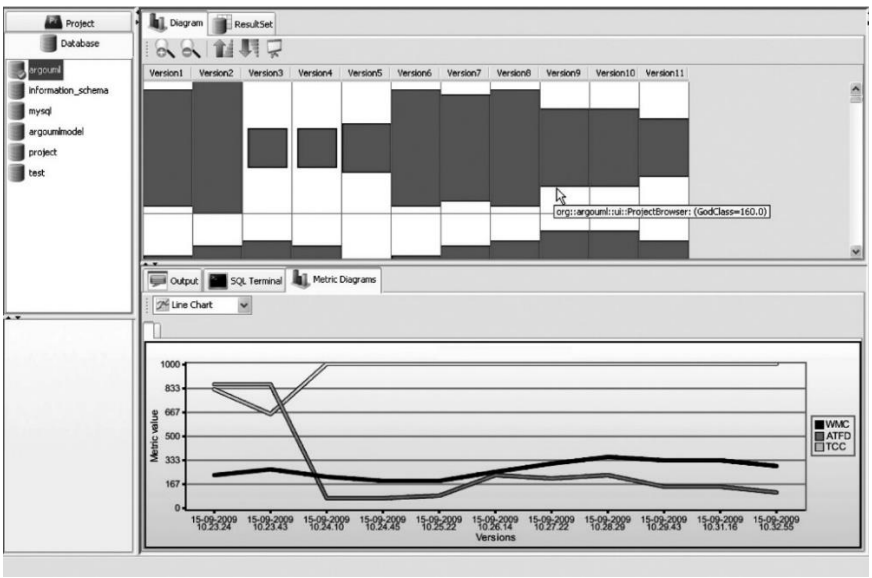

Рис. 3.17. Знімок засобів інтерфейсів користувача

Проведемо аналіз кількості дефектів програмного забезпечення комп'ютерної системи у кожній версії системи. Як показано на рис. 3.18 із оновленням версії спостерігається зростання програмних класів і в тому числі кількості класів уражених дефектами. Під розвитком дефекту слід розуміти, що рівень його розвитку перевищує 100%. Візуально видно, що зростання дефектів є лінійним.

Кількість уражених класів протягом всієї історії залишається відносно стабільною, і тільки в останній версії спостерігається значне погіршення. Крім того, з рис. 3.18 видно, що кількість класів зростає значно швидше, ніж кількість уражених класів, тобто більшість нових класів вносяться без дефектів проектування, а більшість уражених класів створені в першій версії.

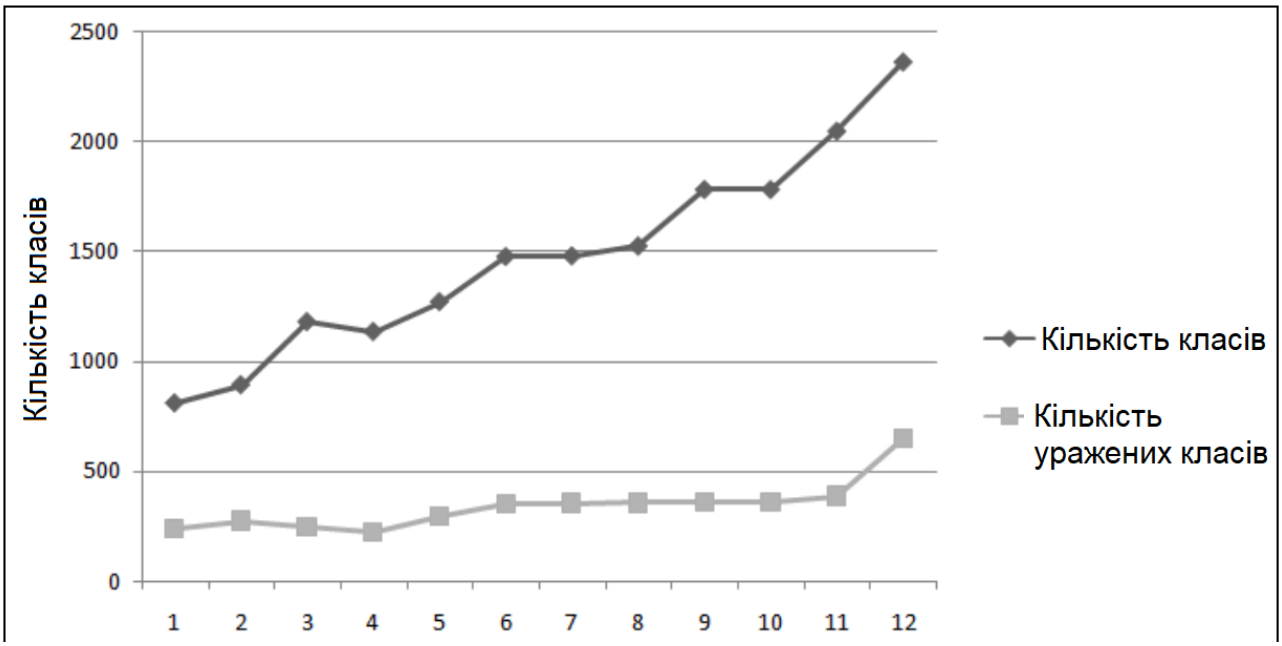

Рис. 3.18. Кількість дефектів у версіях ПЗ КС

Категорії дефектів програмного забезпечення та їх розподіл представлено на рис. 3.19. При цьому 77% дефектів, які наведені в історії, були без змін і формували категорію з найбільш чисельних дефектів.

Рис. 3.19. Категорії дефектів та їх розподіл

Дефекти категорій «Пульсація» і «Зростання» наведені на рис. 3.20 і рис. 3.21 відповідно.

З рис. 3.20 видно, що категорія з найбільшою кількістю дефектів – нижня. У ній містяться дефекти, які перебувають на стадії зародження і рівень розвитку яких не перевищує сто відсотків.

Найбільш небезпечні дефекти містяться у категорії з найменшою кількістю дефектів: «Граничне зростання» - 4% (8), «Гранична пульсація» - 4% (16).

Таким чином, в першу чергу рекомендується провести реструктуризацію 24 класів, уражених даними дефектами для зменшення негативного впливу дефектів на надійність комп’ютерної системи.

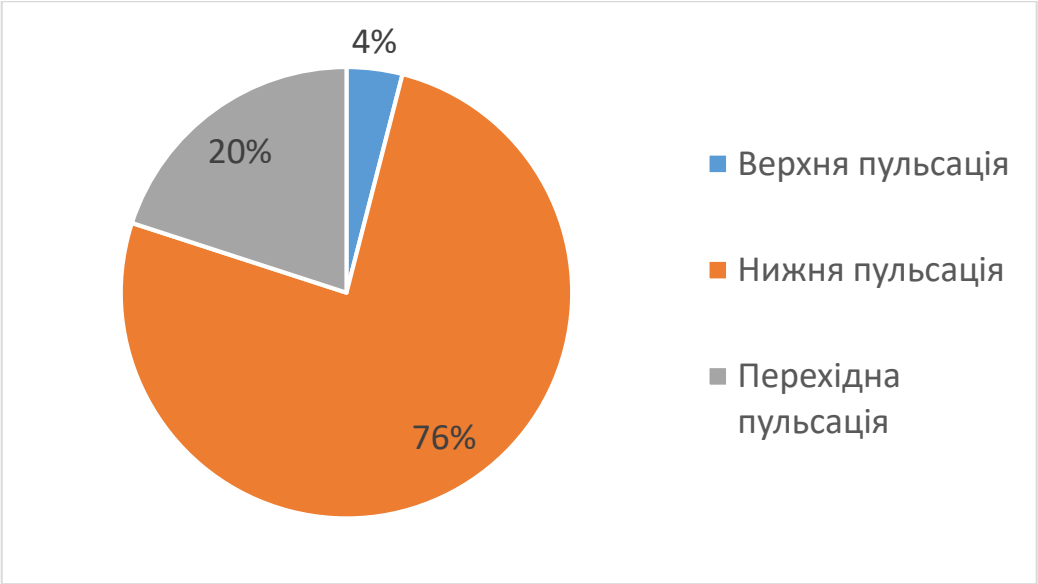

Рис. 3.20. Розподіл дефектів за категорією «Пульсація»

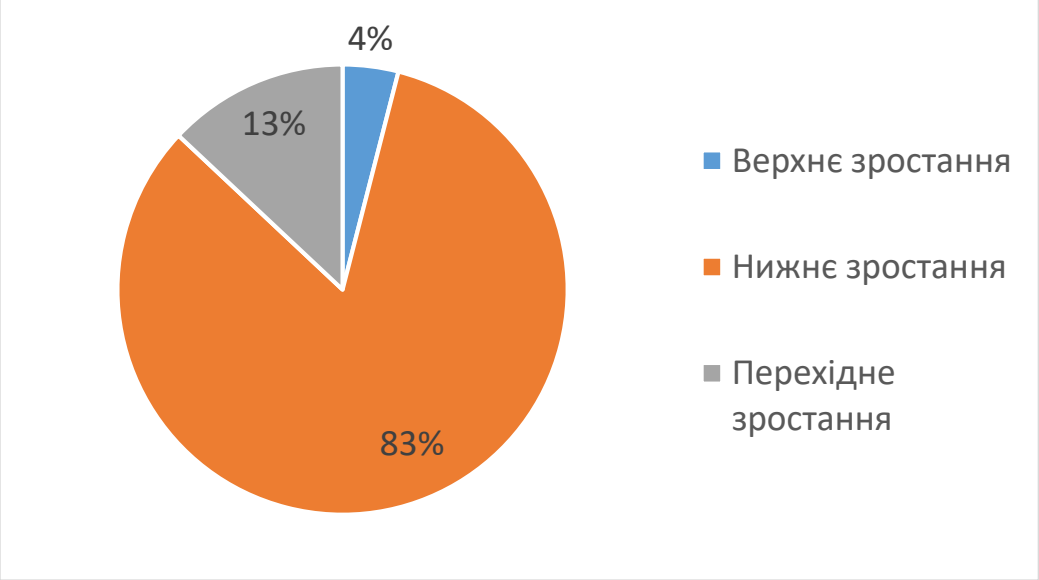

Рис. 3.21. Розподіл дефектів за категорією «Зростання»

На рис. 3.22 показано історію програмного забезпечення комп'ютерної системи. Перший виділений дефект програмного забезпечення (виділення позначено «пульсація») знаходиться у класі ModelElement, оскільки рівень розвитку дефекту, згідно історії перевищує 100% і пульсує.

Рис. 3.22. Історія дефектів програмного забезпечення комп’ютерної системи

Другий виділений дефект (виділення позначено «пульсація») знаходиться в класі FigMessage і є представником категорії «Спадання». З представлення видно, що в другій версії ступінь розвитку даного дефекту перевищив 100%. Однак в наступних версіях розробники, очевидно, провели реструктуризацію, і у восьмій версії дефект був остаточно усунутий. Нарешті, третій виділений дефект (виділення позначено «Зростання») знаходиться в класі PropPanel і є представником категорії «Зростання». Зростання даного дефекту почалося у восьмій версії, після чого ступінь розвитку дефекту стабілізувалася і до останньої версії залишається постійним. Незважаючи на зростання, розглянутий дефект останнім часом стабільний, а значить не викликає проблем при супроводі, тому немає необхідності його усувати, однак при внесенні змін до класу, розробник повинен ретельно їх спроектувати, щоб не допустити прогресу дефекту і уникнути необхідності проводити реструктуризацію в майбутньому.

Моніторинг дефектів проектування дозволяє розпізнавати небезпечні прогресуючі дефекти проектування і своєчасно планувати роботи по їх усуненню, маючи під рукою знімок історії дефектів, розробник може запобігти розвитку дефекту, ретельно проектуючи зміни.

3.6. Висновки до розділу

Основні результати даного розділу полягають в наступному:

1. За допомогою мови моделювання UML та use case діаграм визначено функціональні вимоги до програмного засобу підтримки процесу визначення впливу дефектів програмного забезпечення на надійність комп’ютерних систем, що дало змогу в подальшому врахувати їх при автоматизації процесів оцінювання надійності.

2. На основі реляційного підходу спроектовано базу даних для зберігання та маніпулювання даними при визначенні впливу дефектів програмного забезпечення на надійність комп’ютерної системи та реалізовано її у середовищі MS SQL Server.

3. Спроектовано архітектуру та розроблено інтерфейси користувачів програмного засобу маніпулювання критеріями надійності, моделями прогнозування дефектів програмного забезпечення, формування імовірних шляхів поширення дефектів на компоненти комп’ютерної системи, що дало змогу забезпечити ефективність виявлення та моніторингу дефектів та реалізувати систему засобами мови C# і технології WindowsForm.

4. Проведено експериментальні дослідження щодо застосування запропонованих та обґрунтованих у роботі моделей, методів і програмного засобу, що дало змогу підтвердити доцільність їх використання.

РОЗДІЛ 4
ОБГРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Тема дипломної роботи магістра пов’язана із дослідженням методів і засобів оцінювання надійності комп’ютерних систем при збоях програмного забезпечення. Важливим аспектом впровадження одержаних у роботі результатів є обґрунтування їх економічної ефективності. У даному розділі проведено розрахунки показників економічної ефективності

4.1. Розрахунок норм часу на виконання науково-дослідної роботи

Основні етапи виконання НДР можна визначити наступним чином:

– дослідження наукових публікацій і практик забезпечення надійності комп’ютерних систем;

– дослідження сучасних методів і засобів виявлення та аналізу дефектів програмного забезпечення комп’ютерних систем;

– побудова та обґрунтування моделей представлення дефектів програмного забезпечення;

– побудова та представлення моделей надійності комп’ютерних систем;

– розробка та обґрунтування методу аналізу впливу дефектів програмного забезпечення на надійність комп’ютерних систем;

– розроблення архітектури засобу аналізу дефектів програмного забезпечення комп’ютерних систем;

– створення інструкції з впровадження програмного засобу;

– оформлення інструкцій.

При оцінюванні тривалості виконання окремих робіт використовують нормативи часу або попередній досвід. До таких нормативів відносять тривалість написання операцій (команд), які для окремих підприємств становлять: для однієї операції – 0,5-1,6 год та 8 годин для п'яти операцій (тривалість зміни).

У разі їх відсутності звертаються до експертних оцінок по встановленню тривалості кожного етапу, яка при трьох оцінках обчислюється за формулою [27]

$$T_{вс} = (t_{min} + 4t_{н.й} + t_{max})/6, \quad (4.1)$$

При двох оцінках, експертна оцінка обчислюється за формулою:

$$T_{вс} = (3t_{min} + 2t_{max})/5, \quad (4.2)$$

де $T_{вс}$ – очікуване (середнє) значення тривалості виконання етапу (стадії);

t_{min} – мінімальна оцінка тривалості виконання етапу;

$t_{н.й}$ – найбільш імовірна оцінка тривалості виконання етапу;

t_{max} – максимальна оцінка тривалості виконання етапу.

Дані витрат часу на виконання окремих стадій (етапів) можна звести у табл. 4.1.

Таблиця 4.1

Основні етапи виконання НДР

№ та назва етапу	Середній час виконання стадії (етапу) інженером, год.	
	Інженер	Керівник
1. Дослідження наукових публікацій і практик забезпечення надійності комп'ютерних систем	8	4
2. Дослідження сучасних методів і засобів виявлення та аналізу дефектів програмного забезпечення комп'ютерних систем	16	6

Продовж. табл. 4.1

№ та назва етапу	Середній час виконання стадії (етапу) інженером, год.	
	Інженер	Керівник
3. Побудова та обґрунтування моделей представлення дефектів програмного забезпечення	20	10
4. Побудова та представлення моделей надійності комп'ютерних систем	18	8
5. Розробка та обґрунтування методу аналізу впливу дефектів програмного забезпечення на надійність комп'ютерних систем	28	6
6. Розроблення архітектури засобу аналізу дефектів програмного забезпечення комп'ютерних систем	52	6
7. Створення інструкції з впровадження програмного засобу	14	4
8. Оформлення інструкцій	8	2
Разом	164	46

Витрати часу керівника на виконання окремих стадій (етапів) при недостатній кількості інформації доцільно приймати в межах 5% сумарних витрат часу інженерів на виконання цих стадій (етапів).

4.2. Визначення витрат на оплату праці та відрахувань на соціальні заходи

Відповідно до Закону України «Про оплату праці» заробітна плата – це «винагорода, обчислена, як правило, у грошовому виразі, яку власник або уповноважений ним орган виплачує працівникові за виконану ним роботу».

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника, результатів його праці та господарської діяльності підприємства. Заробітна плата складається з основної та додаткової оплати праці.

Основна заробітна плата нараховується на виконану роботу за тарифними ставками, відрядними розцінками чи посадовими окладами і не залежить від результатів господарської діяльності підприємства.

Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов'язані з виплатами за фактично відпрацьований час. Нараховують додаткову заробітну плату залежно від досягнутих і запланованих показників, умов виробництва, кваліфікації виконавців. Джерелом додаткової оплати праці є фонд матеріального стимулювання, який створюється за рахунок прибутку.

Основна з/п складається із прямої з/п і доплати, яка при укрупнених розрахунках становить 25% – 35% від прямої з/п. При розрахунку з/п кількість робочих днів в місяці слід приймати – 21 дні/міс., що відповідає 168 год./міс. Розмір місячних окладів керівника та інженерів слід приймати згідно існуючих на даний час норм. Основна заробітна плата розраховується за формулою:

$$З_{осн.} = T_c \cdot K_Г \qquad (4.3)$$

де T_c – тарифна ставка, грн.;

$K_Г$ – кількість відпрацьованих годин.

Посадові оклади (тарифні ставки) за розрядами Єдиної тарифної сітки визначаються шляхом множення окладу (ставки) працівника 1 тарифного розряду на відповідний тарифний коефіцієнт. У разі коли посадовий оклад (тарифна ставка) визначені у гривнях з копійками, цифри до 0,5 відкидаються, від 0,5 і вище – заокруглюються до однієї гривні. У 2019 році посадові оклади (тарифні ставки) розраховуються згідно з Законом України «Про Державний бюджет України на 2019 рік».

Мінімальна зарплата у 2019 р. складає 4173,00 грн., в погодинному розмірі 25,13 грн., приймемо 100,00 грн. для інженера, для керівника – 140,00 грн.

Для інженера: $З_{осн.} = 100{,}00 \cdot 164 = 16400{,}00$ грн.

Для керівника: $З_{осн.} = 140{,}00 \cdot 46 = 6440{,}00$ грн.

Додаткова заробітна плата становить 10 – 15% від суми основної заробітної плати і обчислюється за формулою

$$З_{дод.} = З_{осн.} \cdot К_{допл.} \quad (4.4)$$

де $К_{допл.}$ – коефіцієнт додаткових виплат (0,1).

Для інженера: $З_{дод.} = 16400{,}00 \cdot 0{,}1 = 1640{,}00{,}00$ грн.

Для керівника: $З_{дод.} = 6440{,}00 \cdot 0{,}1 = 644{,}00$ грн.

Звідси загальні витрати на оплату праці ($В_{ОП.}$) визначаються за формулою

$$В_{ОП.} = З_{осн.} + З_{дод.} \quad (4.5)$$

Для інженера: $В_{ОП.} = 16400{,}00 + 1640{,}00 = 18040{,}00$ грн.

Для керівника: $В_{ОП.} = 6440{,}00 + 644{,}00 = 7084{,}00$ грн.

Таким чином загальна сума становить 25124,00 грн. Крім того, необхідно визначити відрахування на соціальні заходи:

– податок на доходи фізичних осіб: 18% – 4522,32 грн.;

– військовий збір: 1,5% – 376,86 грн.;

– єдиний внесок: 22% – 5527,28 грн.

У сумі зазначені відрахування становлять 41,5%. Отже, загальна сума відрахувань на соціальні заходи становитиме:

$$В_{С.З.} = ФОП \cdot 0{,}415 \quad (4.6)$$

де $ФОП$ – фонд оплати праці, грн.

У даному випадку сума відрахувань становить: $B_{С.З.} = 25124{,}00 \cdot 0{,}415 = 10426{,}46$ грн.

Проведені розрахунки витрат на оплату праці зведемо у табл. 4.2.

Таблиця 4.2

Зведені витрати на заробітну плату

Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Нарахування на ФОП, грн.	Всього витрати на оплату праці, грн.
	Тарифна ставка, грн.	К-сть відпрацьованих годин	Фактично нарах. з/пл., грн.			
Інженер	100	164	16400,00	1640,00	7486,60	25526,60
Керівник проекту	140	46	6440,00	644,00	2939,86	10023,86
Разом			22840,00	2284,00	10426,46	35550,46

4.3. Розрахунок витрат на електроенергію

Затрати на електроенергію при використанні обладнання визначаються за формулою:

$$З_е = W \cdot T \cdot S \qquad (4.7)$$

де W – необхідна потужність, кВт;

T – кількість годин роботи обладнання;

S – вартість кіловат-години електроенергії.

Згідно з постановою НКРЕКП України від 05.10.2018 р. № 1177 вартість електроенергії становить 243,71 коп./кВт.год.

Потужність комп'ютера – 800 Вт, а кількість годин роботи обладнання згідно табл. 4.1 – 210 годин.

Затрати на електроенергію становлять: $З_е = 0{,}8 \cdot 210 \cdot 2{,}4371 = 409{,}43$ грн.

4.4. Розрахунок витрат на матеріали

Результати розрахунку затрат на матеріали наведено у табл. 4.3.

Таблиця 4.3

Визначення величини затрат на матеріал

Найменування матеріальних ресурсів	Одиниця виміру	Норма витрат	Ціна за одиницю, грн.	Затрати матеріалів, грн	Транспортно – заготівельні витрати, грн.	Загальна сума витрат на матеріали, грн
Середовище MS Visio	шт.	1	6000	6000	-	6000
Папір А4	пачка	2	100	200	-	200
Компакт диски	шт.	2	4	8	-	8
Разом						6208

4.5. Розрахунок суми амортизаційних відрахувань

Характерною особливістю застосування основних фондів у процесі виробництва є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації.

Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їх повного відновлення.

Комп’ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Для визначення амортизаційних відрахувань застосовуємо формулу:

$$A = \frac{Б_в \cdot Н_А}{100\%} \tag{4.8}$$

де A – амортизаційні відрахування за звітний період, грн.,

$Б_в$ – балансова вартість комп’ютера, на початок звітного періоду, грн..

$Н_А$ – норма амортизації, яку приймемо на рівні 15%.

Амортизаційні відрахування при балансовій вартості ПК у 16000 грн. та нормі амортизації на рівні 15%, амортизаційні відрахування становитимуть: $A = \frac{21000 \cdot 15\%}{100\%} = 3150{,}00$ грн.

4.6. Обчислення накладних витрат

Накладні витрати пов’язані з обслуговуванням виробництва, утриманням апарату управління підприємства (фірми) та створення необхідних умов праці.

Накладні витрати можна встановити на рівні 20% від суми основної та додаткової заробітної плати працівників:

$$Н_В = В_{ОП} \cdot 0{,}2 \tag{4.9}$$

де $Н_В$ – накладні витрати, грн.,

$В_{ОП}$ – суми основної та додаткової заробітної плати працівників, грн..

У даному випадку накладні витрати становитимуть:

$Н_В = 25124{,}00 \cdot 0{,}2 = 5024{,}80$ грн.

4.7. Складання кошторису витрат та визначення собівартості науково-дослідних робіт

Собівартість (C_B) науково-дослідних робіт розрахуємо за формулою:

$$C_B = B_{о.п.} + B_{с.з} + З_{м.в.} + З_е + Т_в + А + Н_в \quad (4.10)$$

У даному випадку собівартість (СВ) науково-дослідних робіт розрахуємо за формулою:

$C_B = 25124{,}00 + 10426{,}46 + 6208{,}00 + 409{,}43 + 3150{,}00 + 5024{,}80 = 50342{,}69$ грн

Результати проведених вище розрахунків зведемо у табл. 4.4.

Таблиця 4.4

Кошторис витрат на науково-дослідних робіт

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці (основну і додаткову заробітну плату)	25124,00	49,91%
Відрахування на соціальні заходи	10426,46	20,71%
Матеріальні витрати	6208	12,33%
Витрати на електроенергію	409,43	0,81%
Амортизаційні відрахування	3150,00	6,26%
Накладні витрати	5024,80	9,98%
Собівартість	50342,69	100,00%

4.8. Розрахунок ціни науково-дослідних робіт

Ціну науково-дослідних робіт можна визначити за формулою:

$$Ц = \frac{C_в \cdot (1 + P_{рен.}) + K \cdot B_{н.і.}}{K} \cdot (1 + ПДВ), \quad (4.11)$$

де $P_{рен.}$ – рівень рентабельності, 33 %;

K – кількість замовлень, од. (встановлюється лише при розробці програмного продукту та мікропроцесорних систем);

$B_{н.і.\cdot}$ – вартість носія інформації, грн. (встановлюється лише при розробці програмного продукту);

ПДВ – ставка податку на додану вартість, (20 %).

Оскільки розробка є прикладною, і використовуватиметься тільки для одного підприємства, то для розрахунку ціни не потрібно вказувати коефіцієнти K та $B_{н.і.}$, оскільки їх в даному випадку не потрібно.

Тоді, формула для обчислення ціни розробки буде мати вигляд:

$$Ц = C_в \cdot (1 + P_{рен.}) \cdot (1 + ПДВ) \quad (4.12)$$

Ціна НДР становитиме:

$Ц = 50342{,}69 \cdot (1 + 0{,}33) \cdot (1 + 0{,}2) = 80346{,}94$ грн.

Визначимо величину прибутку:

$$П = Ц - C_в \quad (4.13)$$

Прибуток буде становити: $П = 80346{,}94 - 50342{,}69 = 30004{,}24$ грн.

4.9. Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність (E_p) полягає у відношенні результату виробництва до затрачених ресурсів:

$$E_p = \frac{П}{C_в} \tag{4.14}$$

де $П$ – прибуток;

$C_В$ – собівартість.

Економічна ефективність становить:

$$E_p = \frac{30004,24}{50342,69} = 0,60.$$

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень (T_p):

$$T_p = \frac{1}{E_p} \tag{4.15}$$

В даному випадку термін окупності становить: $T_p = \frac{1}{0,60} = 1,68$ року.

Про доцільність розробки програми можна сказати при врахуванні критеріїв, які наведені у табл. 4.5.

Таблиця 4.5

Техніко-економічні показники НДР

№ з/п	Показник	Значення
1	Собівартість, грн	50342,69
2	Плановий прибуток, грн	30004,24
3	Ціна, грн	80346,94
4	Економічна ефективність	0,596
5	Термін окупності, рік	1,68

Собівартість методу і засобу оцінювання надійності комп’ютерних систем при збоях програмного забезпечення, які одержано при виконанні НДР, становить 50342,69 грн., а термін їхньої окупності – 1,68 року, що дозволяє обґрунтувати економічну доцільність та ефективність впровадження.

РОЗДІЛ 5
ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

5.1. Охорона праці

Тема дипломної роботи магістра пов’язана із дослідженням методів і засобів оцінювання надійності комп’ютерних систем при збоях програмного забезпечення. Такі роботи передбачають використання комп’ютерної техніки на етапах формування пояснювальної записки, налаштування засобів автоматизації процесу оцінювання надійності комп’ютерних систем та виявлення дефектів програмного забезпечення. Тому, при виконанні робіт виявлення впливу помилок у програмному забезпеченні на надійність комп’ютерних систем необхідно враховувати вимоги з охорони праці при експлуатації комп’ютерної техніки.

В Україні діють ряд законів, нормативних документів та актів, які регулюють процеси забезпечення та управління охороною праці у різних галузях народного господарства. До них належать: Конституція України, Закони України "Про охорону праці", "Про охорону здоров'я", "Про пожежну безпеку", "Про використання ядерної енергії та радіаційний захист", "Про забезпечення санітарного та епідемічного благополуччя населення", "Про загальнообов'язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захворювання, які спричинили втрату працездатності", Кодекс законів про працю України (КЗпП).

Однією з основних вимог до приміщень, де робочі місця обладнані комп’ютерною технікою і планується використання програмного комплексу для оцінювання надійності комп’ютерних систем, є вимоги щодо площі, яка відводиться на один ПК. При проектуванні автоматизованих робочих місць інженерів з надійності необхідно дотримуватись вимог щодо розміщення комп’ютерів. На один ПК передбачено площу 6 м^2 та об’єм 20 м^3.

Однак робота з комп’ютером включає різні завдання, які об'єднуються

такими загальними чинниками, як те, що робота проводиться в сидячому положенні і вимагає уважного, неперервного та іноді тривалого спостереження.

Перше правило, якого варто дотримуватись інженерам із забезпечення та оцінювання надійності комп'ютерних систем стосується правильного облаштування робочого столу. При цьому слід передбачити наступні його параметри: фіксована висота – 720 мм, забезпечення необхідного простору для рук по висоті, ширині і глибині, в області сидіння не повинно бути шухляд.

Друге правило визначає облаштування робочого стільця: можливість регулювання висоти стільця, забезпечення обертання конструкції стільця.

У приміщеннях з ПК, на яких планується виконання задач з оцінювання надійності комп'ютерних системи у випадку збою програмного забезпечення, яскравість знаків і яскравість фону дисплею повинна бути спроектована таким чином, щоб не було великої відмінності з яскравістю навколишнього середовища, але знаки повинні чітко пізнаватися на відстані читання.

Характеристики освітлення, зокрема у приміщеннях, де експлуатується ПК, повинні відповідати ДБН В.2.5-28-2006 "Природнє і штучне освітлення". Основні вимоги даного нормативного документу стосуються забезпечення наступних вимог:

- освітлення по можливості із сторони, зліва;
- по можливості - рівномірне освітлення всього робочого простору;
- прилади по можливості встановлюють у місцях, віддалених від вікон;
- встановлення непрямого штучне освітлення;
- світло, що поступає через вікна, «пом'якшують» за допомогою штор;
- робоче місце організовується так, що напрям погляду був паралельний фронту вікон.

Останнє правило, якого необхідно дотримуватись інженерам з надійності комп'ютерних систем, передбачає оптимальний метод роботи, що полягає у передбачені зміни завдань і навантажень, дотримання перерви в роботі: 5 хвилин через 1 годину роботи біля дисплея або 10 хвилин після 2-х годин роботи біля дисплея. Вимоги цього правила регламентовані нормативним документом

ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин»

При створенні сприятливих умов для підвищення продуктивності і зменшення напруги значну роль грають чинники, що характеризують стан навколишнього середовища: мікроклімат приміщення, рівень шуму і освітлення.

Рекомендована величина відносної вологості, яка повинна бути забезпечена у приміщеннях з експлуатації програмного комплексу оцінювання надійності комп'ютерних систем, повинна відповідати ДСанПіН 3.3.2.007-98 і становити 65 – 70%. При цьому робоче місце повинно бути добре вентильованим. У даний час з погляду шумового навантаження досягнутий значний прогрес. Рівень шуму в приміщеннях (приблизно 40 Дб) не перевищує допустимого рівня, незалежно від кількості використовуваної апаратури.

Для приміщень, в яких експлуатується програмний комплекс підтримки запропонованих методів оцінювання надійності комп'ютерних систем у вмпадку збою програмного забезпечення, потрібно забезпечити виконання вимог пожежної безпеки, які визначені Правилами пожежної безпеки в Україні, НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» [24].

Будівлі і ті їх частини, в яких розташовуються ПК можуть належати до II ступеня вогнестійкості. Над та під приміщеннями, де розташовуються ПК, а також у суміжних з ними приміщеннях не дозволяється розташування приміщень категорій А і Б за вибухопожежною небезпекою.

Приміщення категорії В повинні бути відділеними від приміщень з ПК протипожежними стінами.

Таким, чином при дослідженні методів і засобів оцінювання надійності комп'ютерних систем при збоях програмного забезпечення, встановлено, що найбільш повним нормативним документом щодо охорони праці користувачів ПК, до яких належать інженери із забезпечення надійнсоті КС, є НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями». Дотримання вимог, які неведені у цьому документі,

сприяє зниженню негативного впливу ПК, його компонентів та інших зовнішніх пристроїв на інженерів, які проводять роботи щодо забезпечення та оцінювання надійності комп’ютерних систем.

5.2. Застосування режимів захисту і хімічного контролю при викиді НХР у випадку аварії на хімічно небезпечному об’єкті

На підприємствах хімічної, деревообробної, нафтопереробної, харчової промисловості можливе виникнення аварійних ситуацій з викидом сильнодіючих отруйних речовин (СДОР). Причинами таких ситуацій може бути порушення правил експлуатації, вимог правил безпеки. В Україні є 877 хімічно небезпечних об'єктів, них 39 розташовані на території Львівської області. Нарощення хімічного виробництва призвело до зростання кількості промислових відходів, які становлять небезпеку для оточуючого середовища і людей. Тільки токсичних відходів в Україні накопичено більше 4 млрд. т, при середньорічному утворенні 103 млн. т. Проблема безпеки населення в зонах можливого хімічного зараження займає важливе місце в переліку завдань щодо захисту людей у надзвичайних ситуаціях"?

Аварії на хімічно небезпечних об'єктах мають свої особливості до яких, зокрема, відносяться:

- неможливість прогнозування аварії у часі.
- велика ймовірність важких наслідків для життя і здоров'я людини.
- складнощі завчасного вжиття ефективних захисних заходів.

Непередбачуваність економічних і екологічних наслідків тощо. У надзвичайних ситуаціях з потенційно небезпечними хімічними речовинами важливе значення має розуміння властивостей СДОР. Найрозповсюдженішими і небезпечними речовинами, що використовуються у промисловості і побуті, є аміак і хлор.

Аміак – за звичайних умов – це газ, легший за повітря, який легко зріджується під тиском, а при випаровуванні поглинає тепло - сильно

охолоджується. Ця властивість використовується у промислових та побутових холодильниках на м'ясокомбінатах, молокозаводах, овочевих базах, тобто там, де є необхідність в охолодженій продукції. Крім того, він є сировиною багатьох хімічних виробництв. Аміак зберігається і транспортується у зрідженому стані. Як рідина, він легший за воду, має меншу густину і при виході на повітря утворює слабкий дим. Вогненебезпечний, створює вибухові суміші з повітрям, отруйний. Особливо небезпечний для очей. При малих концентраціях діє збуджуючи, при великих – людина непритомніє. Крім того, він викликає задуху, сильний кашель. Найкращі методи захисту – ізолюючий протигаз, респіратор РПГ-67КД, захисний костюм типуЛ-1, гумові чоботи, рукавички. Оскільки аміак легший за повітря, то він буде здійматися вгору, тому безпечніше від аміачної хмари ховатися у низинах, підвалах, тунелях.

Хлор – отруйний, негорючий жовто-зелений газ, зі специфічним запахом хлорки, отрутніший за аміак у 20 разів. Хлор – газоподібний, він трохи важчий за повітря, легко зріджується під тиском. Тому зберігають його і транспортують у сталевих балонах або цистернах. У рідкому стані він важчий за воду. При випаровуванні утворює білий туман. Розчинний у воді, але гірше за аміак.

Хлор широко розповсюджений промисловий продукт, використовується для знезараження питної води, відбілювання тканин, як сировина ця багатьох хімічних підприємствах. У зв'язку з таким способом його використання трапляється чимало випадків отруєння. Так, наприклад, у Брукліні (район Нью-Йорка), коли хлор з віддаленого магнієвого заводу накопичувався у станцію підземки, від нього постраждало понад тисячі чоловік. Це приклад того, що хлор може пересуватися низинами на значні відстані. При концентрації хлору у повітрі понад 0,2 мг/л може статися миттєва смерть. При потраплянні його на шкіру виникають опіки. Як запобігти ураженню хлором? Найкраще використовувати ізолюючий протигаз, кисневий ізолюючий прилад, спеціальний захисний костюм, умові чоботи, рукавиці. За відсутності індивідуальних засобів у нагоді може стати одяг з цупкої тканини, протигаз з

активованим вугіллям. А якщо і цього немає, то слід вдихати повітря через хустинку, змочену розчином соди чи антихлору (розчин фітофіксину з содою). Можна також вмочити тканину сечею, яка частково знешкоджує хлор, або простою ізодою. Слід пам'ятати, що хлор накопичується у низинах, тому треба підніматися догори. На жаль, як показала практика, солдати під час хлорної газової атаки у першу світову війну ховалися, навпаки, у підвалах, землянках та окопах, що значно збільшило кількість жертв та ефективність хімічної зброї.

При отруєнні хлором рекомендується вдихати пари спирту та ефіру, але перед цим постраждалим необхідно забезпечити свіже повітря. При відсутності дихання слід зробити штучне дихання.

Ступінь хімічної небезпеки населення при аваріях з виходом (СДОР) залежить від масштабу аварії, властивостей СДОР, стану атмосфери, рельєфу місцевості тощо. У системі цивільної оборони розроблена «Методика прогнозування масштабів зараження СДОР при аваріях». Вона дозволяє розраховувати можливі площі хімічного зараження та визначати втрати людей. Унаслідок аварій на об'єктах, які виробляють СДОР, обслуговуючий персонал і населення, яке мешкає поблизу об'єкта, можуть отримати тяжкі ураження.

Велике значення має своєчасне та якісне проведення розвідки осередку ураження. Цю роботу ведуть підрозділи хімічної розвідки Збройних сил, ЦО та інші. Вони визначають місце аварії та вид СДОР, ступінь зараження місцевості, шляхи безпечного виходу з неї, беруть проби ґрунту, води тощо і відправляють їх у лабораторію. На початку виникнення і проникнення СДОР в атмосферу або на місцевості негайно оповіщають робітників і службовців об'єктів і населення, яке мешкає поблизу зони, про небезпеку. Люди, які є в будинках, зачиняють вікна, проводять повну герметизацію житла, вимикають нагрівальні прилади, газ. Евакуація населення з районів можливого зараження СДОР проводиться до підходу зараженої хмари. На об'єкті, де була аварія, в першу чергу здійснюється робота з припинення викиду СДОР. Ураженим надається медична допомога. Краплини СДОР на одязі знешкоджують за допомогою індивідуального протихімічного пакета ІПП-8. При роботах в

осередках ураження СДОР треба дотримуватись правил безпеки. Всі люди повинні мати протигази, індивідуальні засоби захисту шкіри, вміти користуватись індивідуальними протихімічними пакетами ІПП-8, а також індивідуальними аптечками АІ-2, вміти надавати першу медичну допомогу.

5.3. Джерела виникнення шуму і вібрацій. Заходи і засоби захисту від шуму і вібрацій, гігієнічні та допустимі норми

При виконанні робіт дипломного проектування необхідно дотримуватись встановлених та затверджених положень з охорони праці і техніки безпеки. Важливим також є дотримання адміністрацією вимог до приміщень, де експлуатується електронно-обчислювальна техніка, і забезпечення необхідних умов для збереження здоров'я та життя людей, надійності та зручності експлуатації апаратного забезпечення.

Оскільки, функціонування розробленої системи, передбачає використання комп’ютерних мереж як способу передачі даних, тому повинні бути дотримані вимоги безпеки при їх експлуатації. Безпека при використанні елементів комп’ютерної мережі, зокрема, ПК та периферійних пристроїв, забезпечується:

– вибором безпечних принципів дії, конструктивних схем, елементів конструкції;

– використанням засобів механізації, автоматизації та дистанційного керування;

– застосуванням в конструкції засобів захисту;

– дотриманням ергономічних вимог;

– включенням вимог безпеки в технічну документацію з монтажу, експлуатації, ремонту та транспортування і зберігання обладнання;

– застосуванням в конструкції відповідних матеріалів.

Дотримання цих вимог в повному обсязі можливе лише на стадії проектування. Тому у всіх видах проектної документації передбачаються

вимоги безпеки. Вони містяться в спеціальному розділі технічного завдання, технічних умов та стандартів на обладнання, що випускається.

Важливими з точки зору збереження життя та здоров’я осіб, які працюють з електронно-обчислювальною технікою є вимоги до шуму та вібрацій, які виникають при роботі з комп’ютерною технікою.

Шум несприятливо діє на слуховий аналізатор та інші органи та системи організму людини. Визначальне значення щодо такої дії має інтенсивність шуму, його частотний склад, тривалість щоденного впливу; індивідуальні особливості людини, а також специфіка виробничої діяльності. Ті види діяльності, у яких поєднується напружена розумова робота та інтенсивне використання комп'ютера (редагування тексту, верстка оригіналу, "запуск" та відлагодження програм, адміністрування тощо) характеризується відчутним впливом навіть незначних рівнів шуму. Цей вплив виражається у зниженні розумової працездатності, швидкій втомлюваності, послабленні уваги, появі головного болю та ін.

Рівні звукового тиску в октавних смугах частот, рівні звуку та еквівалентні рівні звуку на робочих місцях, де використовується комп’ютерна та оргтехніка визначені ДСанПіН 3.3.2-007-98.

Основними заходами та засобами боротьби з шумом є:

– зниження рівнів шуму в джерелі його утворення (застосовується, як правило, в процесі проектування);

– використання звукопоглинаючих та звукоізолюючих засобів;

– раціональне планування виробничих приміщень та робочих місць.

На комп'ютеризованих робочих місцях основними джерелами шуму є вентилятори системного блоку, накопичувачі, принтери ударної дії. Для зниження рівнів шуму на робочих місцях рекомендується розмістити друкувальні пристрої ударної в іншому приміщенні, або огородити їх звукоізолюючими екранами [13].

Оскільки зовнішні шуми (вулиця, суміжні приміщення) також можуть негативно впливати на функціональний стан осіб, які працюють з

комп’ютерною технікою, то стіни приміщень, в яких розташовані комп'ютеризовані робочі місця бажано облицювати звукопоглинаючими матеріалами. Однак доцільність їх застосування повинна бути обґрунтована спеціальними інженерно-акустичними розрахунками. Звукопоглинаюче облицювання (іноді й стелі) необхідно здійснювати матеріалами, що мають максимальний коефіцієнт звукопоглинання в межах частот 31,5-8000 Гц і дозволені для оздоблення приміщень органами державного санітарно-епідеміологічного нагляду [33].

Під час виконання робіт у виробничих приміщеннях з комп’ютерною технікою значення характеристик вібрації на робочих місцях не повинні перевищувати допустимих значень, визначених СН 3044-84 та ГОСТ 12.1.012-90.

Для зниження вібрації обладнання, пристрої, пристосування необхідно встановлювати на спеціальні амортизуючі прокладки, передбачені нормативними документами.

Провівши аналіз специфіки роботи за ПК, вимог до виробничого шуму та вібрацій, правил техніки безпеки та охорони праці з компонентами розробленого програмного засобу, можна зробити висновок про те, що означені вимоги є дотриманими і шкідливість впливу негативних факторів на здоров’я людини зведені до мінімуму.

РОЗДІЛ 6
ЕКОЛОГІЯ

6.1. Формування бази статистичних даних в екології

Для формування бази статистичних даних в екології необхідно одержати інформацію про характеристики досліджуваних явищ чи об’єктів і відповідні встановити їх значення. Основою будь-якої статистичної обробки та оцінки екологічної інформації є збір інформації, що здійснюється методом статистичного спостереження [27].

Статичне спостереження – одна з найважливіших стадій статистичного дослідження. Воно вважається фундаментом статистичного дослідження, адже в процесі його здійснення формується первинна статистична інформація, яка на наступних етапах дослідження підлягає обробленню і аналізу. При цьому відбувається накопичення бази статистичних даних [27].

Статичне спостереження в екології, згідно [27], це планомірний, науково-організований збір масових даних про екологічні явища і процеси. Здійснюється шляхом реєстрації за заздалегідь розробленою програмою спостереження.

Об'єктом спостереження є стан забруднення навколишнього середовища (природних об’єктів) атмосферного повітря, природних водних об’єктів, земель та ґрунтів. Збір даних проводиться не стихійно, а регулярно, що дає змогу вивчити тенденції, напрями, закономірності розвитку екологічних явищ і процесів та зберегти їх у базі даних екологічної інформації [27]. План статистичного спостереження передбачає широке коло питань методики та організації збору статистичної інформації, контролю її якості та вірогідності, які весь час поповнюють базу статистичних даних в екології. Для об'єкта статистичного спостереження характерне те, що його не можна вивчати безпосередньо в цілому, для цього потрібно виділити в його складі окремі одиниці. Одиниця статистичного спостереження – це складовий елемент об'єкта дослідження, який є основою рахунку і носієм істотних ознак та властивостей,

які підлягають реєстрації. Це первинний елемент об'єкта дослідження. Одиницю спостереження встановлюють, виходячи із завдань спостереження і складності об'єкта дослідження [27].

Правильне визначення одиниці спостереження має істотне значення для організації і проведення статистичного дослідження.

Інформація статистичного спостереження повинна бути об'єктивною і якісною, а отже, забезпечуватись правильною науковою організацією її одержання, належним виконанням самого спостереження. Завдання статистичного спостереження зумовлюється завданнями, які ставляться перед дослідженням певних екологічних процесів і явищ і випливають з потреб управління ними [27].

Наукова організація статистичного спостереження, у відповідності до [27], зумовлює дотримання певних вимог щодо його здійснення:

– статистичне спостереження повинно здійснюватися на науковій основі заздалегідь розробленою програмою, яка забезпечувала б науковий підхід до вирішення методологічних та організаційних питань;

– статистичне спостереження повинне забезпечувати збір масових даних, у яких відбивається вся сукупність фактів. Неповнота зведень про досліджувані процеси призведе до помилкових висновків з результатів аналізу;

– орієнтація статистичного спостереження на збирання не тільки інформації, яка безпосередньо характеризує досліджуваний об'єкт, а й такої, що сприяє зміні його стану;

– інформація, одержана за результатами статистичного спостереження, повинна бути вірогідною;

– дані статистичного спостереження повинні бути порівнювані. Лише в такому разі забезпечується їх узагальнення і зіставлення у просторі й часі.

Таким чином, організація статистичного спостереження передбачає формування бази статистичних даних, вимоги до якої повинні враховувати вимоги до процесу самого спостереження.

Програма статистичного спостереження представляє собою перелік питань, на які треба одержати відповіді в процесі збирання статистичних зведень щодо кожної досліджуваної одиниці [27].

Один і той самий об'єкт може бути обстежений з різних боків. Тому склад і зміст питань програми спостереження залежить від завдань дослідження і особливостей об'єкта. Вона повинна охоплювати широке і повне коло відомостей. Чим ширша програма, тим повніше висвітлюється досліджуване явище.

У статистичний практиці застосовують дві організаційні форми спостереження: звітність і спеціально організовані статистичні спостереження.

Звітність – це форма статистичного спостереження, при якій статистичні дані надходять у статистичні органи від підприємств і установ у вигляді обов'язкових і таких, що мають юридичну силу звітів про їх роботу. Таким чином формується база даних звітної інформації.

Звітність підприємств, установ та організацій є поки що основним джерелом статистичної інформації для формування відповідних баз даних. У ній передбачається система строго регламентованих показників, які характеризують діяльність підприємств, установ та організацій. Зміст звіту, форма і термін подання також встановлюється вищим статистичним органом. Звітність складають на основі документів первинного оперативно- технічного і бухгалтерського обліку. Вірогідність гарантується також юридичною відповідальністю керівників підзвітних підприємств та організацій.

За різними ознаками статистичну звітність поділяють на окремі види. Насамперед розрізняють типову і спеціалізовану звітність [27]:

– типова звітність має єдину форму і зміст для всіх підприємств окремої галузі або всього народного господарства.

– спеціалізована звітність властива тим підприємствам чи окремим виробництвам, що мають свої специфічні особливості.

Вид звітності впливає на техніку збору і зведення статистичної інформації, а також на структуру бази даних екологічної інформації. Удосконалення статистичної звітності на сучасному етапі відбувається у напрямі скасування термінової звітності та скорочення кількості поштових звітів.

6.2. Використання в Україні альтернативних джерел енергії

В Україні загальний річний технічно досяжний енергетичний потенціал альтернативних джерел енергії в перерахунку на умовне паливо становить близько 63 млн. тон [28].

Частка енергії добутої за рахунок альтернативних джерел становить сьогодні близько 3%. Згідно з українською енергетичною стратегією до 2030 р. частку альтернативної енергетики на загальному енергобалансі країни буде доведено до 20 %.

Основними та найбільш ефективними напрямами відновлюваної енергетики в Україні є: вітроенергетика, сонячна енергетика, біоенергетика, гідроенергетика, геотермальна енергетика [28].

Таблиця 6.1

Прогнозні показники розвитку використання нетрадиційних та відновлювальних джерел енергії (НВДЕ) за основними напрямами освоєння, млн умовного палива тон/рік

Напрями освоєння НВДЕ	Рівень розвитку НВДЕ по роках			
	2005 р.	2010 р.	2020 р.	2030 р
Позабалансові джерела енергії, всього	13,85	15,96	18,5	22,2
У тому числі шахтний метан	0,05	0,96	2,8	5,8
Відновлювальні джерела енергії, всього	1,661	3,842	12,054	35,53

Продовж. табл. 6.1

Напрями освоєння НВДЕ	Рівень розвитку НВДЕ по роках			
	2005 р.	2010 р.	2020 р.	2030 р
У тому числі: Біоенергетика	1,3	2,7	6,3	9,2
Сонячна енергетика	0,003	0,032	0,284	1,1
Мала гідроенергетика	0,12	0,52	0,85	1,13
Геотермальна енергетика	0,02	0,08	0,19	0,7
Вітроенергетика	0,018	0,21	0,53	0,7
Енергія довкілля	0,2	0,3	3,9	22,7
Всього	15,51	19,83	30,55	57,73

В останні роки, за рахунок прийнятих рішень на законодавчому рівні в Україні активно розвиваються проекти відновлюваної енергетики. Закон України «Про внесення змін щодо встановлення «зеленого тарифу» був прийнятий ще 25 вересня 2008 р. «Зеленим» називається завищений тариф, за яким регульований державою оптовий ринок купує електроенергію у компаній і приватних домогосподарств, що виробляють її з відновлювальних джерел енергії.

За підсумками 2014 р. в Україні вітровими, сонячними та електростанціями, які працюють на біомасі, було вироблено 1,7 млрд кВт/год електроенергії. Ще 9 млрд кВт/год забезпечили гідроелектростанції (ГЕС і ГАЕС). У загальній кількості їх частка склала близько 11,8 % виробництва електроенергії в Україні. Однак частина станцій, які використовують відновлювані джерела електроенергії, залишилися в Криму або на окупованих територіях Донбасу, а саме декілька великих сонячних електростанцій, а також вітропарки у Приазов'ї та Луганській області [28].

За даними Всеукраїнської енергетичної асамблеї, станом на 1 квітня 2015 р. в Україні потужність об'єктів відновлюваної енергетики, яким встановлено «зелений» тариф», становить 1469,21 МВт. Загалом за перше півріччя 2015 р.

обсяг енергії, виробленої завдяки використанню відновлювальних джерел енергії, сумарно склав понад 550 тис. кВт/год або 1,3 % від загального обсягу генерації енергії в Україні [28].

Станом на 2015 р. в нашій державі налічується 142 сонячні електростанції, 21 промисловий об'єкт, що використовує енергію вітру для генерації електроенергії, 105 малих гідроелектростанцій, для яких, на відміну від великих ГЕС і ГАЕС, встановлений «зелений» тариф, а також 5 об'єктів, що працюють на біомасі.

В останні роки найбільш активно розвивалися сонячні електростанції (СЕС), не в останню чергу завдяки високим «зеленим» тарифам». До найбільших підприємств цієї галузі слід віднести ТОВ «Токмак Солар Енерджі», яке було створене за кошти Європейського геоекологічного фонду в рамках Європейської екологічної програми. Однак на початку літа «зелені» тарифи були переглянуті. Так, ціна на енергію, вироблену на СЕС, булла знижена з 50 до 20 євроцентів за кВт/год. Незважаючи на істотне зниження тарифу для СЕС, привабливість інвестицій у цей сегмент залишиться досить високою, оскільки за останні 5 років вартість обладнання для сонячної енергетики скоротилася в кілька разів. Це пов'язано як із розвитком технологій, так і з перевиробництвом фотоелементів у Південно-Східній Азії. Крім того, в деяких країнах ЄС «зелений» тариф для СЕС вже скасовано [28].

Перспективним для України є використання енергії вітру, тому великі компанії продовжують інвестувати кошти у цей бізнес. Зокрема, компанія «Вінд Пауер», що володіє найбільшою в Україні Ботієвською вітроелектростанцією (інвестиції у її будівництво склали 340 млн євро), має намір продовжити будівництво вітроелектростанцій у Запорізькій області. Протягом двох років може також відновитися будівництво Приморської вітроелектростанції потужністю 200 МВт.

Виробництво електроенергії на малих гідроелектростанціях, попри дещо вищу, порівняно з великими ГЕС, собівартість електроенергії дає змогу економити значні обсяги паливно-енергетичних ресурсів. Так, наприклад,

Явірська ГЕС (Львівська обл.) з досить невеликою встановленою потужністю (близько 450 кВт) за рік дозволяє економити 800 т вугілля, яке спалила б теплова електростанція такої ж потужності [28]. Крім того, малі ГЕС не тільки виробляють електроенергію, а й захищають прилеглі населені пункти від повеней, сприяють їх нормальному водопостачанню, розвитку рибного господарства.

Через те, що Україна є переважно рівнинною країною, у ній не слід очікувати активного розвитку електроенергетики, яка працює на використанні біомаси. Водночас, у нашій державі розвивається агробізнес, що дає хороші перспективи електростанціям, що використовують біомасу. Саме тому Українська асоціація відновлюваної енергетики наполягала на збільшенні «зеленого» тарифу для цієї сфери, що має стимулювати інвесторів [28].

Окупність проектів у сфері використання відновлювальних джерел енергії, зазвичай, становить від 6 до 12 років і залежить від великої кількості чинників. Експерти стверджують, що у нинішніх умовах поки не слід очікувати буму у відновлюваній енергетиці, оскільки в країні немає коштів розвивати цю галузь і найближчим часом вони не з’являться. Більшість об’єктів відновлюваної енергетики, які сьогодні працюють в Україні, побудовані за рахунок кредитів ЄБРР або експортного фінансування країнами-виробниками основного обладнання. Тим не менш, до 2020 р. Україна зобов’язалася збільшити частку «зеленої» генерації до 11 %. Сьогодні це близько 6 % від загального обсягу генерації електроенергії в країні [28].

ВИСНОВКИ

Основні результати дипломної роботи магістра полягають в наступному:

1. У результаті аналізу базових принципів організації та побудови комп’ютерних систем, встановлено, що одним з найбільш важливих її компонентів є програмне забезпечення, від стабільності і надійності роботи якого залежить ефективність експлуатації комп’ютерної системи.

2. Проведено аналіз помилок програмного забезпечення, їх типів та впливу на надійність, що дало змогу враховувати їх розвиток у комп’ютерній системі та обґрунтувати необхідність застосування моделей прогнозування показників надійності на різних стадіях життєвого циклу і спостереження за ними під час експлуатації і супроводу.

3. Проведено аналіз моделей надійності комп’ютерних систем з акцентом на програмне забезпечення і виявлено, що застосування моделей оцінювання надійності технічних засобів є не ефективним для представлення показників надійності програмного забезпечення у зв’язку з різною природою та механізмами відмов.

4. На основі результатів аналізу моделей раннього і пізнього прогнозування помилок програмного забезпечення обґрунтовано їх застосування при проектуванні комп’ютерних систем, що дає змогу побудувати комплексний підхід до визначення впливу дефектів програмного забезпечення на надійність експлуатації комп’ютерних систем з врахуванням історії виникнення помилок у ПЗ.

5. Визначено та формалізовано показники надійності програмного забезпечення комп’ютерних систем, що дають змогу кількісно виражати наявність, щільність та інтенсивність дефектів, які можуть призводити до збоїв у роботі програмних складових та негативно впливати на надійність комп’ютерної системи в цілому.

6. Побудовано та обґрунтовано модель впливу дефектів програмного забезпечення на надійність комп’ютерної системи, що забезпечує можливість

встановлення шляху поширення дефекту програмного забезпечення на інші компоненти комп’ютерної системи і дає змогу прогнозувати ймовірність виникнення негативного впливу на надійність комп’ютерної системи.

7. Обґрунтовано модель і метод моніторингу дефектів програмного забезпечення комп’ютерних систем на основі модифікованої моделі Hismo, що дає змогу аналізувати розвиток і вплив дефекту програмного забезпечення на надійність комп’ютерної системи у часі та з врахуванням версій програмного забезпечення.

8. Обґрунтовано моделі прогнозування дефектів програмного забезпечення комп’ютерних систем, що дають змогу на «ранніх» і «пізніх» стадіях розробки комп’ютерних систем виявляти можливі загрози збою програмного забезпечення та знижувати їх негативний вплив на надійність комп’ютерної системи.

9. За допомогою мови моделювання UML та use case діаграм визначено функціональні вимоги до програмного засобу підтримки процесу визначення впливу дефектів програмного забезпечення на надійність комп’ютерних систем, що дало змогу в подальшому врахувати їх при автоматизації процесів оцінювання надійності.

10. На основі реляційного підходу спроектовано базу даних для зберігання та маніпулювання даними при визначенні впливу дефектів програмного забезпечення на надійність комп’ютерної системи та реалізовано її у середовищі MS SQL Server.

11. Спроектовано архітектуру та розроблено інтерфейси користувачів програмного засобу маніпулювання критеріями надійності, моделями прогнозування дефектів програмного забезпечення, формування імовірних шляхів поширення дефектів на компоненти комп’ютерної системи, що дало змогу забезпечити ефективність виявлення та моніторингу дефектів та реалізувати систему засобами мови C# і технології WindowsForm.

12. Проведено експериментальні дослідження щодо застосування запропонованих та обґрунтованих у роботі моделей, методів і програмного засобу, що дало змогу підтвердити доцільність їх використання.

13. Обгрунтовано економічну ефективність одержаних наукових і практичних результатів шляхом обчислення показників собівартості (50342,69 грн) і терміну окупності (1,68 року) методу та засобу оцінювання надійності комп'ютерних систем при збоях програмного забезпечення, що дозволяє зробити висновок про доцільність та ефективність їх впровадження.

14. Проаналізовано вимоги з охорони праці і техніки безпеки при роботі з програмним засобом підтримки процесу оцінювання надійності комп'ютерних систем при збоях програмного забезпечення, визначено шляхи мінімізації негативного впливу шкідливих факторів на здоров'я користувачів ПК, а також застосування режимів захисту і хімічного контролю при викиді НХР у випадку аварії на хімічно небезпечному об'єкті.

15. Проведено аналіз методів і засобів формування бази статистичних даних в екології та використання в Україні альтернативних джерел енергії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сергієнко І.В. Становлення і розвиток досліджень з інформатики. К.: Наукова думка. 2008. 205 с.

2. Лаврищева Е.М. Основы технологической подготовки разработки прикладных программ систем обработки данных. АН УССР, Ин-т кибернетики им. В.М. Глушкова; 87-5. Киев. 1987. 30 с.

3. ДСТУ 3918-99 Інформаційні технології. Процеси життєвого циклу програмного забезпечення. Київ. Держстандарт України. 2000. 49 с.

4. IEEE 982.1 Standard Dictionary of Measures to produce Reliable Software. IEEE. 1988. 56 p.

5. Highsmith J. Retiring Lifecycle Dinosaurs. Software Testing & Quality Engineering. 2010. URL: http: //www.stqemagazine.com (дата звернення 08.11.2019 р.)

6. Андон Ф.И., Лаврищева Е.М. Методы инженерии распределенных компьютерных систем. К.: Наукова думка. 1997. 228 с.

7. Лаврищева Е.М., Грищенко В.Н. Сборочное программирование. К.: Наукова думка. 1991. 213 с.

8. Лаврищева Е.М. Сборочное программирование. Некоторые итоги и перспективы. Проблемы программирования. №2. 1999. С. 20 – 31.

9. Липаев В.В. Надежность программных систем. М.: СИНТЕГ.1998. 321с.

10. Коваль Г.И., Коротун Т.М. Проблемы анализа причинно-следственных связей отказов и ошибок в ПО. Сб. Программная инженерия. Киев: Ин-т кибернетики им. В.М. Глушкова НАНУ. 1994. С. 83 - 93.

11. Коваль Г.И., Коротун Т.М., Яблокова Т.Л., Куцаченко Л.И. Планирование обеспечения надежности информационных систем. Проблемы программирования.№3-4. 2001. С. 40 - 47.

12. Lakey P.B., Neufelder A.M. System and software reliability assurance notebook. Rome Laboratory Report, Griffiss Aif Force Base, Rome NY. 2017. 186 p.

13. Мороз Г.Б., Коваль Г.И., Коротун Т.М. Концепция профилей в инженерии надежности программных систем. Математичні машини і системи. №1. 2004. С. 166 – 184.

14. Musa J.D. Operational Profiles in Software Reliability Engineering. IEEE Software.V.10. N.2. 2003. P. 14 - 32.

15. Munson J., Elbaum S. Software reliability as a function of user execution patterns. Proc. of the 32nd Hawaii International Conference on System Sciences. 1999. P.1-12.

16. Hecht H. An Alternative Software Reliability Assessment. Proceedings 14h International Symposium on Software Reliability Engineering (ISSRE 2003), Denver, Colorado. 2003. P. 293 - 295.

17. Ohlsson N., Helander M., Wohlin C. Quality Improvement by Identification of Fault-Prone Modules using Software Design Metrics. Proceedings Sixth International Conference on Software Quality. 1996. P. 1 – 13.

18. Коваль Г.И. Прогнозирование надежности ПО компьютерных систем. Сб. материалов конференции. УкрПРОГ’98, 2-4 сентября 1998 г. Киев. 1998. С. 358 - 361.

19. Мороз Г.Б., Лаврищева Е.М. Модели роста надежности ПО. Киев. - 1992. 25 с.

20. Fenton N.E. A critique of software defect prediction models. IEEE Trans. On Soft. Eng. V. 25. N.5. 1999. P. 675 – 689.

21. Коваль Г.И. Подход к прогнозированию надежности ПО при управлении проектом. Проблемы программирования. № 1 – 2. 2002. С. 282 – 290.

22. Malaiya Y.K., Denton J. What do the Software Reliability Growth Model Parameters Represent. Proc. IEEE-CS Int. Symp. on Software Reliability Engineering ISSRE. 1997. P. 124 - 135.

23. Chulani S. Constructive quality modeling for defect density prediction: COQUALMO. International Symposium on Software Reliability Engineering (ISSRE'99). Boca Raton. 1999.

24. Бойчик І. М. Економіка підприємства: Навч. посібник. К.: Атіка, 2004. 480 с.

25. Жидецький В.Ц. Охорона праці користувачів комп’ютерів. Львів: Афіша. 2000. 176с.

26. Желібо Є, Заверуха Н., Зацарний В. Безпека життєдіяльності. К.: 2001. 483 с.

27. Тарасова В.В. Екологічна статистика. Київ: «Центр учбової літератури», 2008. 391с.

28. Дудченко О. Альтернативні джерела енергоресурсів в Українському Причорномор'ї. Аналітична записка. URL: http://od.niss.gov.ua/articles/232/ (дата звернення 16.11.2019 р.).

Додаток А

Текст публікації

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний технічний університет імені Івана Пулюя (Україна)
Національна академія наук України
Університет імені П'єра і Марії Кюрі (Франція)
Маріборский університет (Словения)
Технічний університет у Кошице (Словаччина)
Вільнюський технічний університет ім. Гедимінаса (Литва)
Шяуляйська державна колегія (Литва)
Жешувський політехнічний університет ім. Лукасевича (Польща)
Білоруський національний технічний університет (Республіка Білорусь)
Міжнародний університет цивільної авіації (Марокко)
Національний університет біоресурсів і природокористування України (Україна)
Наукове товариство ім. Шевченка
ГО «Асоціація випускників Тернопільського національного технічного університету імені Івана Пулюя»

АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ

Збірник

тез доповідей

Том ІІ

VIII Міжнародної науково-технічної конференції молодих учених та студентів

27-28 листопада 2019 року

УКРАЇНА
ТЕРНОПІЛЬ – 2019

УДК 004.05
Є.В. Тиш, канд. техн. наук, Я.І. Юськів
Тернопільський національний технічний університет імені Івана Пулюя, Україна

МОДЕЛЬ ВИЯВЛЕННЯ ВПЛИВУ ДЕФЕКТІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА НАДІЙНІСТЬ КОМП'ЮТЕРНИХ СИСТЕМ

Ie.V. Tysh, Y.I. Yuskiv

MODEL OF SOFTWARE DEFECTS INFLUENCE DETECT ON RELIABILITY OF COMPUTER SYSTEMS

Сучасні комп'ютерні системи представляють собою сукупність програмно-апаратних засобів, які взаємодіють через визначені канали зв'язку і забезпечують необхідний рівень функціональності, надійності, продуктивності, зручності використання та інших характеристик. При цьому складність, розподіленість та багатозадачність комп'ютерних систем обумовлює посилення вимог до надійності їхнього функціонування у заданому контексті використання.

Оскільки, до складу комп'ютерних систем входить апаратне забезпечення, вбудоване та високорівневе програмне забезпечення, то важливими задачами в галузі забезпечення та моніторингу надійності комп'ютерних систем є дослідження впливу помилок, внесених у компоненти системи.

До складу типової комп'ютерної системи входять: апаратна складова, програмне забезпечення різних рівнів та канали передачі даних, тому представимо комп'ютерну систему у вигляді сукупності:

$$CompSyst = \{ HW, SW, Chan \} \quad (1)$$

$CompSyst$ - комп'ютерна система;

HW – сукупність апаратного забезпечення;

SW – сукупність програмного забезпечення;

$Chan$ – сукупність каналів передачі даних.

В загальному випадку, надійність комп'ютерної системи залежить від надійності її компонентів і виражається як зважена сума показників надійності компонентів формули (1).

Для представлення надійності програмного забезпечення можна записати наступний вираз

$$R(SW) = \{ SW_i, Def_{ij} \} \quad (2)$$

$R(SW)$ – надійність програмного забезпечення комп'ютерної системи;

SW_i – компонент програмного забезпечення, $i = 1..k$, k – кількість компонентів програмного забезпечення;

Def_{ij} – дефект, пов'язаний з функціонуванням i-го компоненту програмного забезпечення, $j = 1..m$, m – кількість дефектів.

$$R(HW) = \{ HW_i, Def_{ij} \} \quad (3)$$

$R(HW)$ – надійність апаратного забезпечення комп'ютерної системи;

HW_i – компонент апаратного забезпечення, $i = 1..l$, l – кількість компонентів

апаратного забезпечення;

Def_{ij} – дефект, пов'язаний з функціонуванням i-го компоненту апаратного забезпечення, $j = 1..p$, p – кількість дефектів.

З іншого боку при проектуванні архітектури комп'ютерної системи, в тому числі й архітектури програмного забезпечення, встановлено зв'язки між компонентами чи модулями системи, які представляють метрику зчеплення між модулями. Для представлення зв'язків між компонентами програмного забезпечення можна записати

$$Rel_{sw} = \{ SW_i, SW_j \} \quad (4)$$

SW_i – i-ий компонент програмного забезпечення комп'ютерної системи;

SW_j – j-ий компонент програмного забезпечення комп'ютерної системи, $i, j \in K, i \neq j$.

Для апаратного забезпечення комп'ютерної системи зв'язки між ними можна представити по аналогії до (4)

$$Rel_{HW} = \{ HW_i, HW_j \} \quad (5)$$

HW_i – i-ий компонент апаратного забезпечення комп'ютерної системи;

HW_j – j-ий компонент апаратного забезпечення комп'ютерної системи, $i, j \in L, i \neq j$

Для представлення зв'язків між апаратними компонентами комп'ютерної системи і програмними складовими запишемо

$$Rel_{HW}^{SW} = \{ \{ SW_i, HW_j \}, Chan_s \} \quad (6)$$

Rel_{HW}^{SW} – множина зв'язків між програмним і апаратним забезпеченням комп'ютерної системи;

$\{ SW_i, HW_j \}$ – пара «програмне забезпечення-апаратний пристрій», які взаємодіють між собою, $i \in K, j \in L$;

$Chan_s$ – один з каналів обміну даними між програмним і апаратним забезпеченням, $s = 1..S$, S – кількість каналів передачі даних.

Таким чином, за допомогою побудови ланцюга взаємодії програмного і апаратного забезпечення комп'ютерних систем, можна визначити потенційний вплив дефектів програмного забезпечення, що призводять до збоїв, на надійність комп'ютерної системи, яка описуються такими ж атрибутами надійності що й програмне забезпечення.

Для підвищення надійності функціонування комп'ютерних систем пропонується мінімізувати зчеплення між модулями як всередині програмного забезпечення, так і зв'язків між програмними і апаратними складовими, шляхом використання підходів функційного програмування.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

VII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ »

11–12 грудня 2019 року

ТЕРНОПІЛЬ
2019

УДК 004.05
Я. Юськів, Є. Тиш
Тернопільський національний технічний університет імені Івана Пулюя

БАЗА ДАНИХ ПІДТРИМКИ ПРОЦЕСУ ОЦІНЮВАННЯ ВПЛИВУ ДЕФЕКТІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА НАДІЙНІСТЬ КОМП'ЮТЕРНИХ СИСТЕМ

UDC 004.05
Ya. Yuskiv, Ie. Tysh
(Ternopil I.Pulyu National Technical University, Ukraine)

DATABASE OF SUPPORTING SYSTEM OF INFLUENCE SOFTWARE DEFECTS ON THE RELIABILITY OF COMPUTER SYSTEMS

Для проектування бази даних підтримки процесу оцінювання впливу дефектів програмного забезпечення на надійність комп'ютерних систем пропонується скористатись підходом реляційних баз даних.

При цьому таблиці будуть відображати сутності предметної області, стовпці у таблицях – властивості сутностей.

При проведенні аналізу предметної області та процесів оцінювання впливу дефектів програмного забезпечення на надійність комп'ютерних систем визначено 8 сутностей:

- «Комп'ютерна система» (Computer System);
- «Програмні складові КС» (Software);
- «Апаратні складові КС» (Hardware);
- «Канали зв'язку»(Channel);
- «Метрика» (Metrics);
- «Дефект» (Defect);
- «Оцінка дефекту» (Evaluation);
- «Критерії надійності»(ReliabilityCriteria).

На рис. 1 наведено ER-діаграму спроектованої бази даних.

Рис. 1. ER-діаграма бази даних

Схему бази даних нормалізовано та приведено до третьої нормальної форми, що забезпечує цілісність та не значну надлишковість даних. Наступний етап полягає у проектуванні архітектури програмного засобу підтримки процесу визначення впливу дефектів програмного забезпечення на надійність комп'ютерної системи.

Додаток Б

Скрипт генерації бази даних

```
CREATE DATABASE REALIBILITY

CREATE TABLE Computer_System (
ID_CS int not null primary key identity (1,1),
CSName varchar (MAX),
[Version] varchar(20),
UpdatingDate Date
)
CREATE TABLE Software (
ID_SW int not null primary key identity (1,1),
ID_CS int foreign key references Computer_System (ID_CS),
ComponentName varchar (100),
[Type] varchar (25),
InputData varchar (MAX),
OutputData varchar (MAX),
ComponentVersion varchar (25),
UpdatingDate Date
)

CREATE TABLE Hardware (
ID_HW int not null primary key identity (1,1),
ID_CS int foreign key references Computer_System (ID_CS),
HWName varchar (100),
[Type] varchar (25),
InputData varchar (MAX),
OutputData varchar (MAX),
UpdatingDate Date
)

CREATE TABLE Channel (
ID_Ch int not null primary key identity (1,1),
ID_CS int foreign key references Computer_System (ID_CS),
ChName varchar (100),
[Type] varchar (25),
[Version] varchar (25),
Protocol varchar (30),
[Description] varchar(MAX)
)

CREATE TABLE Metric (
ID_M int not null primary key identity (1,1),
MetricName varchar (100),
Formula varchar (100),
[Description] varchar(MAX)
)

CREATE TABLE RealibilityCriteria (
ID_RC int not null primary key identity (1,1),
CriteriaName varchar (100),
Formula varchar (100),
```

```
[Description] varchar(MAX)
)

CREATE TABLE Defect (
ID_Def int not null primary key identity (1,1),
DefectName varchar (100),
DefectNum int,
ID_SW int foreign key references Software (ID_SW),
ID_HW int foreign key references Hardware (ID_HW),
ID_Ch int foreign key references Channel (ID_Ch),
ID_M int foreign key references Metric (ID_M),
Value float,
[Description] varchar(MAX)
)

CREATE TABLE Evaluation (
ID_Ev int not null primary key identity (1,1),
ID_def int foreign key references Defect (ID_Def),
[Type] varchar (35),
Mark float,
Acceptability bit default (0),
MaxValue varchar (50),
MinValue varchar (50),
[Description] varchar(MAX)
)
```