Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: «Розробка програмного забезпечення для перевірки сумісності ліцензій Java-проєктів»

Виконав: студент IV курсу, групи СПс-41
спеціальності 121 – Інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) Гудима І.А.
(прізвище та ініціали)

Керівник (підпис) Коноваленко І.В..
(прізвище та ініціали)

Нормоконтроль (підпис) Стоянов Ю.М.
(прізвище та ініціали)

Завідувач кафедри (підпис) Петрик М.Р.
(прізвище та ініціали)

Рецензент (підпис)
(прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
проф. Петрик М.Р.
(підпис) (прізвище та ініціали)
« 06 » квітня 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Гудимі Ігорю Анатолійовичу

1. Тема роботи Розробка програмного забезпечення для перевірки сумісності ліцензій Java-проектів

Керівник роботи Коноваленко Ігор Володимирович, к.т.н., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом по університету від « 04 » квітня 2026 року № 4/9-172

2. Термін подання студентом роботи 22.06.2026

3. Вихідні дані до роботи наукові літературні джерела

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1 Огляд предметної галузі.
2. Проєктування та розробка плагіна.
3. Інтерфейс користувача плагіна. Тестування.
4. Безпека життєдіяльності, основи охорони праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема роботи. 2. Актуальність, мета, задачі дослідження
3. Сумісність популярних відкритих ліцензій.
4.Порівняння можливостей популярних інструментів. 5. Архітектура плагіна.
6. Функціональні вимоги. 7. Програмні засоби та технології.
8. Алгоритм контролю ліцензій. 9. Процес пошуку потенційних порушень ліцензування.
10. Вкладки Project License та Package License.
11. Панель редактора для роботи з файлом основної ліцензії проєкту.
12. Редультати тестування плагіна.
13. Висновки по роботі.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці			

7. Дата видачі завдання 06.04.2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз предметної галузі дослідження	06.04–17.04.26	
2.	Обґрунтування актуальності дослідження	18.04–26.04.26	
3.	Огляд існуючих механізмів та засобів	27.04–04.05.26	
4.	Проведення дослідження механізмів та засобів		
	опрацювання даних	05.05–10.105.26	
5.	Оформлення розділу «Огляд предметної галузі»	11.05–14.05.26	
6.	Оформлення розділу «Проєктування та розробка		
	плагіна»	15.05–18.05.26	
7.	Оформлення розділу «Інтерфейс користувача плагіна.		
	Тестування»	19.06–23.05.26	
8.	Оформлення розділу «Безпека життєдіяльності,		
	основи охорони праці»	20.05–25.05.26	
9.	Нормоконтроль	06.06–15.06.26	
10.	Перевірка на плагіат	12.06–16.06.26	
11.	Попередній захист роботи	15.06–20.06.26	
12.	Захист кваліфікаційної роботи	27.06.26	

Студент _____ Гудима І.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Коноваленко І.В..
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка програмного забезпечення для перевірки сумісності ліцензій Java-проєктів // Кваліфікаційна робота освітнього рівня «бакалавр» // Гудима Ігор Анатолійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра програмної інженерії, група СПс-41 // Тернопіль, 2026 // С. - 54, табл. - 2, рис. - 18, слайд. – 13, додат. - 1 , бібліогр. – 40.

Ключові слова: відкритий код, ліцензія, модуль, порушення, сумісність, Java- проєкт, JetBrains

У першому розділі роботи здійснено огляд предметної області, в тому числі наведено можливості open-source програмного забезпечення. Розглянуто різні типи можливих ліцензії програмних продуктів, досліджена їх сумісність. Описано можливі порушення ліцензування відкритого програмного забезпечення. Досліджено характеристики наявних на ринку програмних засобів, котрі застосовуються для одержання інформації щодо ліцензій Java-проєктів.

У другому розділі представлена розроблена архітектура плагіна. Концептуально він складається із двох модулів. Наведено алгоритм контролю ліцензій, втілення котрого гарантує вилучення і виокремлення інформації щодо дозволених ліцензій проєкту. При функціонуванні плагіна втілена можливість надання програмісту інформації про можливі порушення у ліцензійному забезпеченні.

У третьому розділі наведено можливості розробленого графічного інтерфейсу плагіна та представлена апробація його роботи. Проведене тестування підтвердило, що створений плагін може успішно використовуватися з метою відшукання потенційних порушень чи ліцензування.

Четвертий розділ висвітлює важливі питання безпеки життєдіяльності та основ охорони праці.

ABSTRACT

Software development for checking the compatibility of Java project licenses // Bachelor Thesis // Hudyma Ihor // Ternopil Ivan Puluj National Technical University, Computer and Information Systems and Software Engineering Faculty, Department of Software Engineering, group SPs-41 // Ternopil, 2026 // P. - 54, tab. - 2, fig. - 18, slid. - 13, annexes - 1, references - 40.

Keywords: open source, license, module, violation, compatibility, Java project, JetBrains

The first section of the Thesis provides an overview of the subject area, including the capabilities of open-source software. Various types of possible licenses for software products are considered, their compatibility is investigated. Possible violations of open-source software licensing are described. The characteristics of software tools available on the market that are used to obtain information about Java project licenses are investigated.

The second section presents the developed architecture of the plugin. Conceptually, it consists of two modules. A license control algorithm is presented, the implementation of which guarantees the extraction and separation of information about the permitted licenses of the project. When the plugin operates, the possibility of providing the programmer with information about possible violations in the licensing is implemented.

The third section presents the capabilities of the developed graphical interface of the plugin and presents the testing of its work. The testing conducted confirmed that the created plugin can be successfully used to find potential violations or licensing.

The fourth section highlights important issues of life safety and the basics of occupational health and safety.

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

ПЗ – програмне забезпечення.

IDE (англ. Integrated Development Environment) – інтегроване середовище розробки.

OSI (англ. Open Source Initiative) – некомерційна організація, створена для просування відкритого програмного забезпечення.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. Зазвичай програмісти схильні повторно використовувати будь-який код, що знаходиться у відкритому доступі, не звертаючи уваги на його ліцензію. У великих IT- компаніях питанням ліцензування займаються спеціальні юристи, які перевіряють ліцензії відкритого ПЗ, що використовується в продуктах компанії, і виявляють можливі несумісності. Водночас у звичайних розробників часто не вистачає ресурсів для таких пильних перевірок.

Для того, щоб вивчити поточну ситуацію із запозиченням коду та порушенням ліцензій, лабораторія Методів машинного навчання в програмній інженерії JetBrains Research провела дослідження щодо потенційних порушень ліцензування в популярних Java -проектах на GitHub. В рамках дослідження було вивчено та описано значну кількість відкритих ліцензій, а також складну систему їх сумісності. За результатами дослідження виявилося, що цілих 9,4% методів Java -класів у даних проектах потенційно могли бути скопійовані з порушенням ліцензування.

Щоб допомогти розробникам не робити подібних помилок і не витрачати час на окремі перевірки, можна показувати на запит необхідну інформацію прямо в IDE. Така можливість дозволить визначати сумісні ліцензії проекту та повідомляти програм місту про потенційні порушення. Завдяки цьому розробники зможуть уникати помилок, пов'язаних з ліцензіями, що потенційно знизить кількість порушень ліцензування у відкритих програмних продуктах

Метою роботи є розробка плагіна до IntelliJ IDEA для роботи з ліцензіями Java- проєктів. Плагін повинен дозволити на підставі ліцензій компонентів проекту визначати допустимі ліцензії для кожного модуля проекту та повідомляти програміста про потенційні порушення.

Були сформульовані завдання, виконання яких необхідне для досягнення поставленої мети:

– провести огляд предметної галузі;

– проаналізувати існуючі рішення у цій галузі;

– розробити архітектуру плагіна;

– реалізувати алгоритм контролю ліцензій проєкту;

— реалізувати процедуру отримання інформації про ліцензії компонент проєкту;

– реалізувати механізм визначення допустимих ліцензій для модуля проєкту на підставі ліцензій компонентів проєкту;

– реалізувати інтерфейс користувача плагіна;

– провести апробацію створеного інструменту.

1 ОГЛЯД ПРЕДМЕТНОЇ ГАЛУЗІ

В даний час є значна кількість open source projects - проєктів із відкритим вихідним кодом. У 2025 році у відкритому доступі на популярному хостингу IT-проєктів GitHub [1] існувало понад 190 000 відкритих проєктів [2], і їх кількість продовжує зростати.

Відкриті ліцензії ПЗ дозволяють розробникам належним чином використовувати, модифікувати та розповсюджувати ПЗ. За порушення умов ліцензії передбачено відповідальність, що включає великі грошові штрафи. Ліцензування відкритого ПЗ – складна область з безліччю нюансів. На даний момент налічується біля 500 різних відкритих ліцензій [3].

Відповідно до результатів дослідження фон Крога та ін. [4] існують дві основні причини порушення ліцензій ПЗ з відкритим вихідним кодом. Перша причина полягає в тому, що через обмеження в ресурсах і часі розробники часто повторно використовують сторонній код, щоб не витрачати сили на вирішення тривіальних завдань. Друга причина полягає в тому, що через велику кількість відкритих ліцензій розробники ПЗ часто не розуміють їх умови та відмінності між ними, що згодом може призводити до порушень ліцензування [5].

1.1 Відкрите ПЗ

Відкрите ПЗ (open source software) — це ПЗ з відкритим вихідним кодом. Розробка перших відкритих проєктів почалася наприкінці 70-х - початку 80-х рр. 20 ст.. Один з найбільш відомих таких проєктів — це система комп'ютерної верстки TeX , яка розробляється з 1979 року відомим вченим і програмістом Дональдом Кнутом [6]. Іншим проєктом стала операційна система GNU [7], розробка якої почалася в 1980 році відомим програмістом Річардом Столлманом. Однак широку популярність open source рух почав набирати після створення в 1998 році організації OSI [8], яка покликана пропагувати принципи open source розробки та підтримувати відкриті проєкти.

В рамках роботи організації OSI було сформульовано визначення open source ПЗ, яке містить вичерпну інформацію про те, яке ПЗ можна вважати відкритим. З повним текстом визначення можна ознайомитись на офіційному сайті OSI.

ПЗ, як та інші форми інтелектуальної власності, підлягає ліцензуванню. Далі розглянемо ліцензії ПЗ.

1.2 Ліцензії ПЗ

Ліцензія ПЗ — це правовий інструмент, що визначає використання, модифікацію та розповсюдження ПЗ, захищеного авторським правом.

Усі ліцензії ПЗ можна розділити на п'ять різних типів [9], кожен з яких має власний набір дозволів та обмежень використання. Крім них, існує ще статус “громадського надбання”, яке не є ліцензією, але теж є способом надання прав. Різні способи передачі прав представлені на рисунку 1.1.

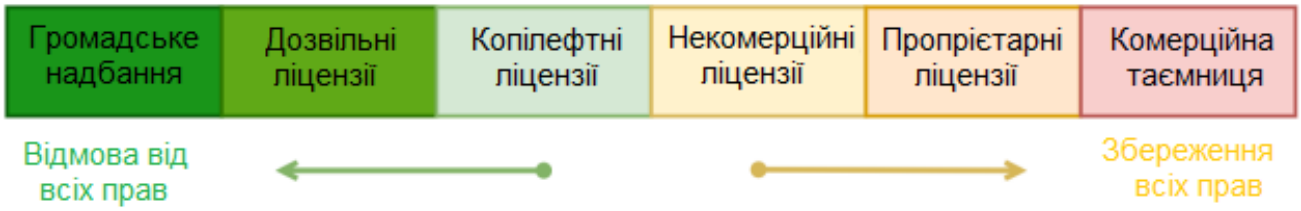

Рисунок 1.1 – Методи передачі прав на ПЗ

Варто зазначити, що ліцензії, що входять до категорій “Дозвільні ліцензії” та “Копілефтні ліцензії” є відкритими (open source), оскільки ПЗ, ліцензоване таким чином , задовольняє визначенню відкритого ПЗ OSI .

Найбільш вичерпний список відкритих ліцензій можна знайти на офіційному сайті відкритого стандарту для передачі відомостей про програмне забезпечення The Software Package Data Exchange (SPDX) [10], котрий містить відомості щодо ліцензій та авторські права .

Стисло розглянемо особливості кожного способу передачі прав.

Громадське надбання. Найбільш дозвільним інструментом для передачі прав

є статус суспільного (громадського) надбання (public domain). Роблячи своє ПЗ громадським надбанням, автори відмовляються від усіх прав і дають можливість розпоряджатися програмним продуктом без будь-яких обмежень. У тому числі, у третіх осіб з'являється можливість зробити це ПЗ закритим або навіть продати його. Поняття суспільного надбання у різних країнах різниться, що може призводити до юридичних труднощів. Щоб уникнути цього, використовуються дозвільні ліцензії.

Дозвільні ліцензії. Дозвільні (permissive) ліцензії надають можливість використовувати, модифікувати та розповсюджувати відповідне ПЗ, включаючи можливість ліцензувати модифікації ПЗ пропрієтарними ліцензіями. Такі ліцензії часто вимагають вказувати оригінальних авторів ПЗ під час поширення модифікованого ПЗ. Крім цього, дозвільні ліцензії часто включають відмову від будь-яких гарантій ПЗ і знімають будь-яку відповідальність з авторів продукту. Найбільш поширеними дозвільними ліцензіями [11] є MIT, Apache 2.0 та BSD-3-Clause.

Іноді автори ПЗ хочуть обмежити можливість ліцензування модифікацій продукту. Щоб це забезпечити , використовуються копілефтні ліцензії.

Копілефтні ліцензії. Зазвичай виділяють два види таких ліцензій: сильні копілефтні (strong copyleft) та слабкі копілефтні (weak copyleft) ліцензії .

Сильні копілефтні ліцензії — це ліцензії, які, як і дозвільні, дозволяють вільно модифікувати та поширювати модифікований продукт, проте ліцензія модифікованого продукту має бути такою самою, як і на оригінальному ПЗ. Сильні копілефтні ліцензії також вимагають вказувати оригінальних авторів на модифікаціях продукту. Таким чином, сильні копілефтні ліцензії дозволяють авторам обмежити використання свого програмного продукту та заборонити використання його модифікацій під іншими ліцензіями. Найбільш популярними сильними копілефтними ліцензіями є ліцензії сімейства GNU General Public License (GPL) [11].

Слабкі копілефтні ліцензії відрізняються від сильних різними послабленнями в обмеженнях використання, модифікації та розповсюдження програмного продукту. Найбільш відомим прикладом таких ліцензій є ліцензії

сімейства GNU Lesser General Public License (LGPL). Вони, як і ліцензії сімейства GPL, дозволяють змінювати та поширювати модифікований код лише за умови збереження цієї ліцензії, проте при використанні коду як бібліотеки виконання цієї вимоги не обов'язкове.

Некомерційні ліцензії. Некомерційні (noncommercial) ліцензії надають права на модифікацію та розповсюдження ПЗ, але тільки для некомерційної діяльності. Часто некомерційні ліцензії є копілефтними ліцензіями. Прикладами таких ліцензій є Java Research License (JRL) та Aladdin Free Public License (AFPL).

Пропрієтарні ліцензії. Пропрієтарні (proprietary) ліцензії — це ліцензії, які забезпечують автору ПЗ монополію на його використання, копіювання та модифікацію. ПЗ, що знаходиться під цим типом ліцензій, є пропрієтарним. Вихідний код таких продуктів зазвичай закритий, але це необов'язково.

Комерційна таємниця. Комерційна таємниця (trade secret) — інформація, котра володіє комерційною цінністю завдяки її невідомості для третіх осіб, до якої відсутній вільний доступ на законній підставі. ПЗ, що є комерційною таємницею, ніде не публікується. Вихідний код таких продуктів зберігається в таємниці, а за розголошення інформації про внутрішню будову ПЗ, що підпадає під комерційну таємницю, передбачено відповідальність.

1.3 Сумісність типів ліцензій

Одним з найважливіших аспектів в галузі ліцензування ПЗ є сумісність різних типів ліцензій. У загальному випадку сумісність типів ліцензій представлено на рисунку 1.2.

Пряма сумісність визначає, якою ліцензією можуть ліцензуватися похідні продукти, які використовують ПЗ автора. Зворотна сумісність визначає, яку ліцензію автор може використовувати для ліцензування свого програмного продукту, виходячи з ліцензій ПЗ, яке він використовував.

Наприклад, у програмному проєкті, який має дозвільну ліцензію, можна використовувати лише ПЗ, що знаходиться під дозвільною ліцензією. У той же час,

модифікації такого проєкту можна поширювати під різними ліцензіями, у тому числі пропрієтарними.

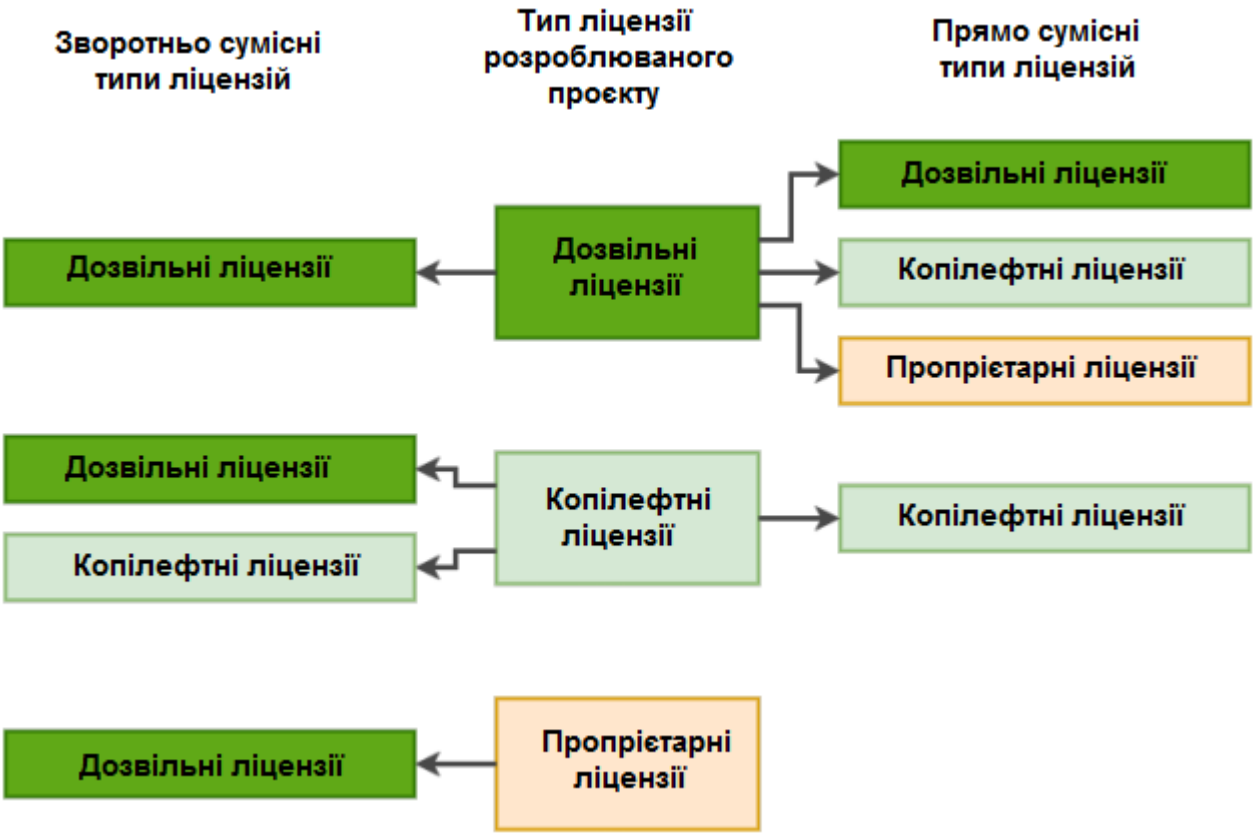

Рисунок 1.2 – Сумісність типів ліцензій

Слід зазначити, що у діаграмі відсутній тип некомерційних ліцензій, оскільки йому не можна загалом визначити сумісність з іншими типами ліцензій. Сумісність з некомерційними ліцензіями необхідно розглядати окремо для кожної ліцензії, тому в рамках даної роботи вони розглядатися не будуть.

1.4 Сумісність відкритих ліцензій

Особливу увагу необхідно приділити сумісності відкритих ліцензій. Важливо відзначити, що використання сторонньої ліцензованої бібліотеки є повторним використанням коду даної бібліотеки , а отже, вимагає дотримання всіх ліцензійних обмежень . Тому одним із завдань плагіна, що розробляється, є пошук потенційних порушень ліцензійних обмежень і повідомлення розробника про знайдені

порушення.

На рисунку 1.3 детально зображено сумісність найбільш популярних відкритих ліцензій.

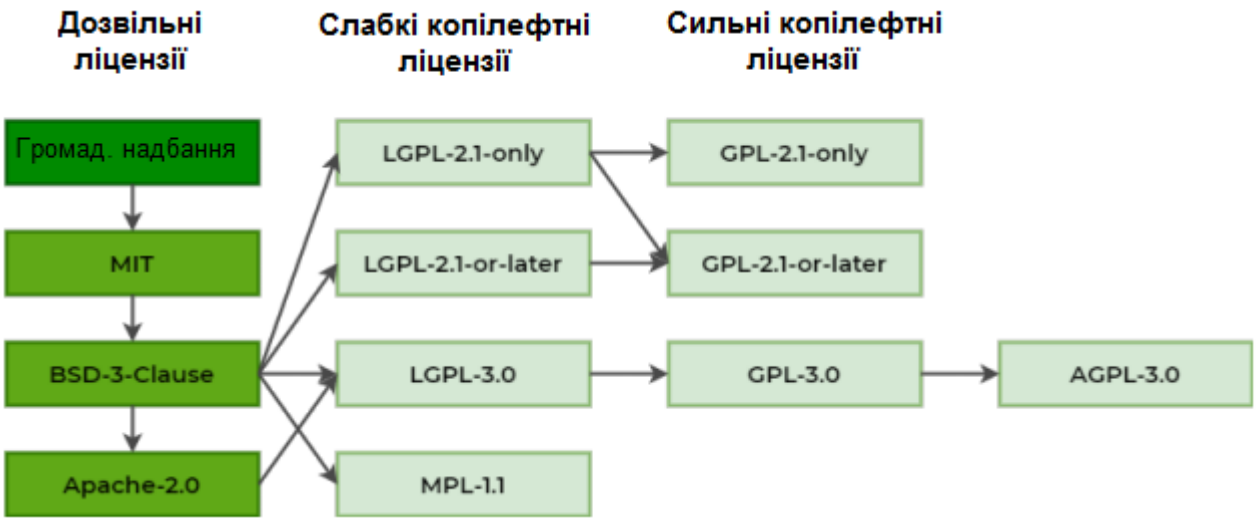

Рисунок 1.3 – Сумісність популярних відкритих ліцензій

Стрілки між ліцензіями позначають їхню пряму сумісність. Важливо, що кожна розглянута на рис. 1.3 ліцензія сумісна сама із собою. Крім цього на діаграмі мається на увазі транзитивна сумісність ліцензій. Наприклад, у продукті з ліцензією GPL-3.0 можна використовувати ПЗ під ліцензією Apache-2.0, але не навпаки.

1.5 Дослідження порушень ліцензування ПЗ

Існує ряд наукових праць, присвячених дослідженню порушень ліцензування у відкритому ПЗ. Одним з таких досліджень є робота Даніеля Германа та ін *[12]*, в результаті якої були виявлені несумісності ліцензій залежностей пакетів дистрибутива Fedora-12 GNU/Linux. Код під ліцензіями GPL часто некоректно використовувався в пакетах з більш рішучими ліцензіями.

У роботі Романського та ін. [13] досліджено можливі порушення ліцензій при міграції коду між модулями Python і постами сервера StackOverflow, а також

зроблено висновок про те, що пости StackOverflow часто містять код, ліцензія якого несумісна з ліцензією плат форми.

Іншою подібною роботою є емпіричне дослідження несумісності ліцензій у відкритому ПЗ Арунеш Матура та ін. [14]. В рамках цього дослідження було вивчено відкриті проєкти хостингу репозиторіїв Google Code, серед яких було виявлено та докладно описано низку проблем, пов'язаних з несумісністю ліцензій.

Хунью Чжан та ін. провели дослідження, присвячене автоматичній перевірці сумісності ліцензій [15], в ході якого був розроблений інструмент для визначення сумісності ліцензій проєктів на базі сервісу Google Code Search. За результатами експериментів з цим інструментом було виявлено декілька нових порушень ліцензій у відкритому ПЗ.

Перелічені дослідження показують, що ліцензування відкритого ПЗ є нетривіальною областю. Незважаючи на те, що написано багато статей і створено цілу низку інструментів для автоматичного визначення ліцензії файлу та виявлення несумісності ліцензій, ця область вимагає подальших досліджень.

1.6 Огляд існуючих технічних рішень

На даний момент існує кілька популярних інструментів, які надають можливість витягувати інформацію про ліцензії з Java-проєктів, включаючи інформацію про ліцензії залежностей проєкту та файлів з вихідним кодом. Такі інструменти корисні для невеликих компаній та незалежних розробників, які не мають достатньо ресурсів для використання послуг професійних юристів з роботи з ліцензіями проєктів.

Після аналізу інструментів для роботи з ліцензіями проєктів були виділені наступні параметри для порівняння:

– компоненти проєкту, що аналізуються інструментом;

– тип інструменту (консольний застосунок, графічний застосунок, веб-сервіс, плагін для системи складання проєкту);

– здатність рекомендації основної ліцензії проєкту;

– здатність визначення потенційних порушень ліцензування;
– відкритість вихідного коду.

Можливість витягувати інформацію не тільки про основну ліцензію проєкту, а й про ліцензії залежностей і файлів з вихідним кодом , безпосередньо визначає можливості самого інструменту.

Тип інструменту визначає зручність його використання. Зазвичай користувачеві простіше працювати з інструментом, який має наочний графічний інтерфейс або інтегрований з популярною системою складання проєктів. Консольні програми, зазвичай, незручні для швидкого освоєння та використання. Часто їхні команди мають безліч параметрів, що потребує часу на їх вивчення і часто відштовхує програмістів від використання таких інструментів.

Незважаючи на те, що список ліцензій усіх компонентів проеєкту сам по собі є корисною інформацією для програміста, йому, як і раніше, необхідно розбиратися у виборі коректної ліцензії для свого проєкту. Інструменти для роботи з ліцензіями проєктів можуть спростити цей процес і надати користувачеві підказки, які допоможуть коректно вибрати ліцензію та уникнути всіх складнощів, пов'язаних з ліцензуванням ПЗ.

Визначення потенційних порушень ліцензування є однією із цілей інструментів для роботи з ліцензіями проєктів. Дана можливість дозволяє попередити програміста про порушення ліцензування при використанні бібліотеки або файлів з вихідним кодом, ліцензії яких не сумісні з основною ліцензією проєкту, що розробляється.

Відкритий вихідний код інструменту дозволяє використовувати його безкоштовно, що є актуальним для незалежних розробників та невеликих компаній. Крім того, у сторонніх розробників з'являється можливість адаптувати код відкритого інструменту під свої потреби, що потенційно збільшує застосовність даного інструменту.

У таблиці 1.1 наведено порівняння деяких популярних інструментів для роботи з ліцензіями Java-проєктів.

Таблиця 1.1 – Порівняння можливостей популярних інструментів

Інструмент	**Ліцензії**	**Тип**	**Рекомендації**	**Порушенн**	**Open**
askalono [16]	Проєкт, вихідний код	Консольний застосунок	-	-	+
FOSSology [17]	Проєкт, залежності, вихідний код, бінарні файли	Веб-сервіс	+	+	-
go-license-detector [18]	Проєкт, вихідний код	Консольний застосунок	-	-	+
license-checker [19]	Проєкт, вихідний код	Консольний застосунок	-	-	+
licensee [20]	Проєкт	Консольний застосунок	-	-	+
Scancode- toolkit [21]	Проєкт, вихідний код, бінарні файли	Консольний застосунок	-	-	+
FOSSA [22]	Проєкт, залежності, вихідний код, бінарні файли	Веб-сервіс	+	+	-
License Maven Plugin [23]	Початковий код	Maven плагін	-	-	+
Gradle License Plugin [24]	Проєкт, залежності	Gradle плагін	-	-	+
Gradle License Report [25]	Залежності	Gradle плагін	-	-	+

Порівняння показало, що більшість інструментів працюють лише з основними ліцензіями проєктів та їх вихідним кодом, не обробляючи ліцензії залежностей проєкту. Жоден із розглянутих безкоштовних інструментів не надає

користувачеві рекомендації щодо вибору основної ліцензії проєкту і не повідомляє про несумісні ліцензії. Крім того, жоден із інструментів не вбудований в IDE.

Інтеграція інструменту для роботи з ліцензіями з популярною IDE полегшує встановлення та налаштування інструменту, а також дає можливість програмісту користуватися таким інструментом, не виходячи з вже звичного редактора коду, що підвищує мотивацію програміста приділяти необхідну увагу ліцензіям проєкту. Завдяки інтеграції інструменту в IDE воно може у фоновому режимі сканувати проєкт на наявність ліцензій та відслідковувати потенційні порушення. Все це дозволить програмісту переконатися, що всі ліцензійні вимоги в проєкті дотримані.

1.7 Функціональні вимоги

За результатами проведених досліджень, викладених у п. 1.6, були сформульовані наступні функціональні вимоги, які ставляться до розроблюваного плагіна:

– контролювати сумісності ліцензії Java-проєкту;

– одержувати інформацію щодо ліцензій компонентів проєкту;

– визначати можливі ліцензії для модуля проєкту на підставі ліцензій його компонентів проєкту;

– формувати файл з найкраще допустимою ліцензією модуля;

– надавати програмісту інформацію щодо можливих порушень у ліцензіях.

1.8 Висновки до першого розділу

В цьому розділі досліджено предметну область, зокрема описано особливості відкритого ПЗ. Розглянуто типи ліцензії ПЗ, їх сумісність. Досліджено можливі порушення ліцензування ПЗ. Проаналізовано можливості існуючих програмних інструментів, які використовуються для отримання інформації про ліцензії Java-проєктів. Сформульовані функціональні вимоги до розробки.

2 ПРОЄКТУВАННЯ ТА РОЗРОБКА ПЛАГІНА

Важливою особливістю розробки даного плагіна є небажаність використання сторонніх щодо компанії JetBrains інструментів для детектування ліцензій проєкту. Тому всі залежності проєкту є продуктами, або внутрішніми розробками компанії JetBrains.

2.1 Архітектура плагіна

Плагін License Detector Plugin (LDetP) для роботи з ліцензіями Java проєктів у IntelliJ IDEA розроблено в рамках проєкту JetBrains Research.

LDetP написаний мовою Kotlin і підтримує роботу з 16 найпопулярнішими ліцензіями Java -проєктів. Плагін складається з двох модулів — Core та Integration. Архітектура плагіна представлена на рисунку 2.1.

Обидва модулі залежать від компоненти IntelliJ Platform SDK [26] — набору інструментів, необхідні створення плагінів на платформі IntelliJ Platform [27].

Перший із модулів, Core, містить внутрішні компоненти, що реалізують алгоритм контролю ліцензій проєкту. Компонент Project Manager відповідає за відстеження стану проєкту та оновлення інформації про знайдені ліцензії компонентів проєкту. License Manager здійснює визначення сумісних ліцензій для кожного модуля проєкту, а також виявляє потенційні порушення ліцензій бібліотек та підмодулів кожного модуля проєкту.

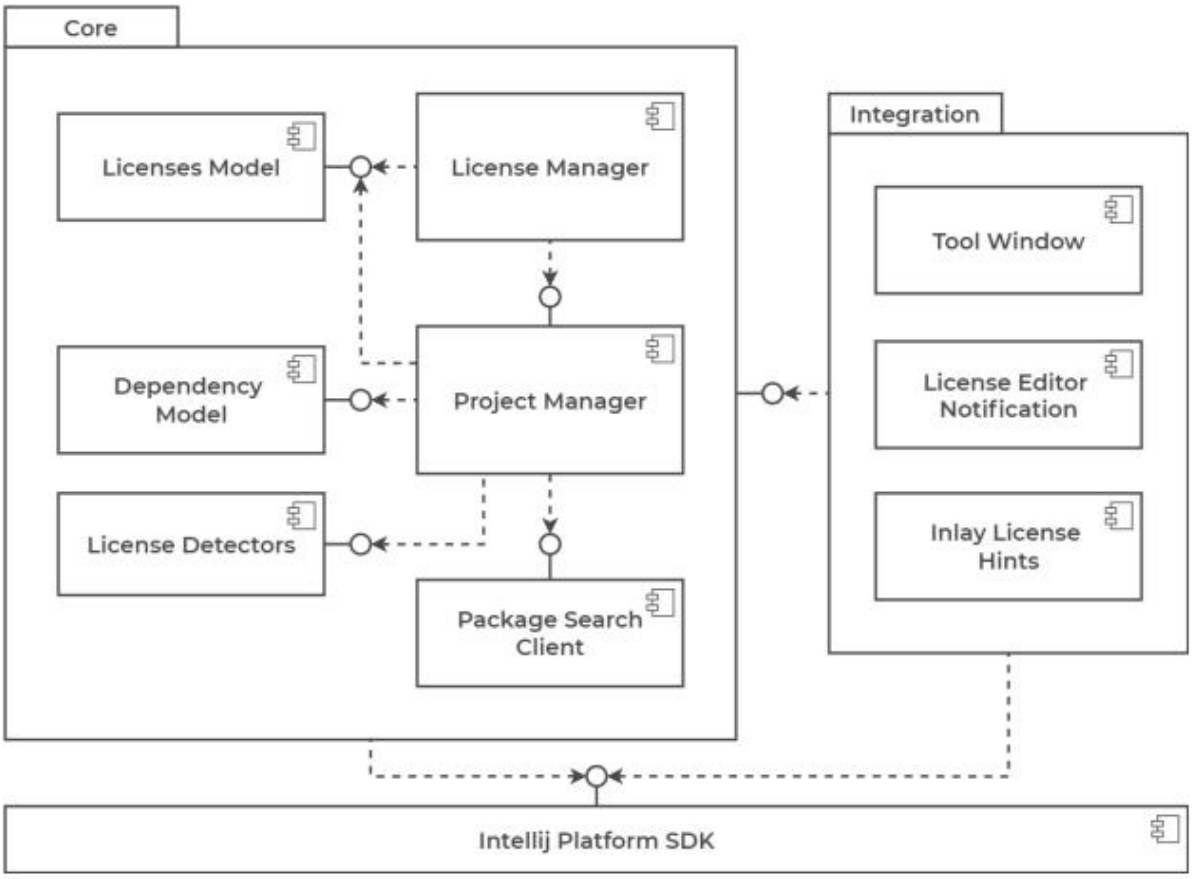

Рисунок 2.1 – Архітектура плагіна

Компонент License Detectors містить детектори ліцензій, які визначають тип ліцензії на основі її повного тексту. Виділення детекторів в окрему компоненту дозволило абстрагуватися від процесу визначення ліцензій при реалізації алгоритму контролю ліцензій проєкту, а також спростило додавання нових детекторів. Package Search Client відповідає за взаємодію з Package Search, сервісом для пошуку Java бібліотек від компанії JetBrains. Компоненти Dependency Model і License Model містять моделі даних залежностей і ліцензій відповідно, що дозволяє уніфікувати роботу з різними залежностями та ліцензіями проєктів, а також знижують накладні витрати на розширення плагіна та підтримку нових ліцензій.

Другий модуль, Integration, відповідає за інтеграцію з IntelliJ IDEA і надає користувачеві графічний інтерфейс, який дозволяє працювати з інформацією про ліцензії у зручному вигляді. Компонент Tool Window відповідає за вікно, що згортається, з інформацією про ліцензії проєкту. License Editor Notification надає панель вгорі редактора файлу з ліцензією, а Inlay License Hints відповідає за

підказки під час редагування скриптів систем для збирання проєктів.

Таким чином, розроблена архітектура дозволяє відокремити логіку роботи з ліцензіями проєкту від інтерфейсу користувача, а також спрощує додавання нових детекторів ліцензій і підтримку нових типів ліцензій у плагіні.

На рисунку 2.2 наведено фрагмент програмного коду.

```
1   import org.jetbrains.changelog.markdownToHTML
2
3   fun properties(key: String) = project.findProperty(key).toString()
4
5   plugins {
6       // Java support
7       id("java")
8       // Kotlin support
9       id("org.jetbrains.kotlin.jvm") version "1.6.10"
10      // gradle-intellij-plugin - read more: https://github.com/JetBrains/gradle-intellij-plugin
11      id("org.jetbrains.intellij") version "1.3.0"
12      // gradle-changelog-plugin - read more: https://github.com/JetBrains/gradle-changelog-plugin
13      id("org.jetbrains.changelog") version "1.1.2"
14  }
15
16  group = properties("pluginGroup")
17  version = properties("pluginVersion")
```

Рисунок 2.2 – Фрагмент вихідного коду

2.2 Алгоритм контролю ліцензій

Алгоритм контролю ліцензій відстежує ліцензії компонента проєкту та визначає допустимі ліцензії для кожного модуля. Допустимі ліцензії модуля — це ліцензії, які міг би мати модуль і які не порушують ліцензії бібліотек та ліцензії інших модулів проєкту. На підставі допустимих ліцензій модулів алгоритм також знаходить потенційні порушення ліцензування у проєкті.

Етапи функціонування алгоритму представлені на рисунку 2.3.

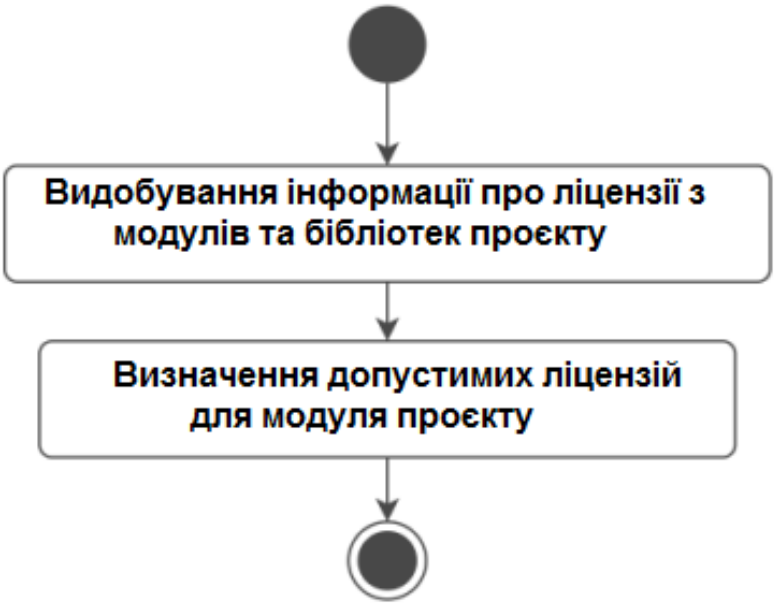

Рисунок 2.3 – Алгоритм контролю ліцензій

Першим етапом є вилучення інформації про ліцензії компонент проєкту. На цьому етапі відбувається сканування модулів і бібліотек проєкту, а також пошук та вилучення інформації про їх ліцензії.

На другому етапі відбувається обробка отриманої інформації про ліцензії модулів та бібліотек, визначаються допустимі ліцензії для кожного модуля проєкту та виявляються потенційні порушення ліцензій бібліотек та підмодулів.

Розглянемо докладніше процес отримання інформації про ліцензії компонент проєкту.

2.2.1 Вилучення інформації про ліцензії компонентів проєкту

Вилучення інформації про ліцензії походить з модулів та бібліотек проєкту. Отримана інформація використовується для визначення допустимих ліцензій для кожного модуля проєкту та пошуку потенційних порушень ліцензування.

Процес отримання інформації про ліцензії компонентів проєкту подано на рисунку 2.4.

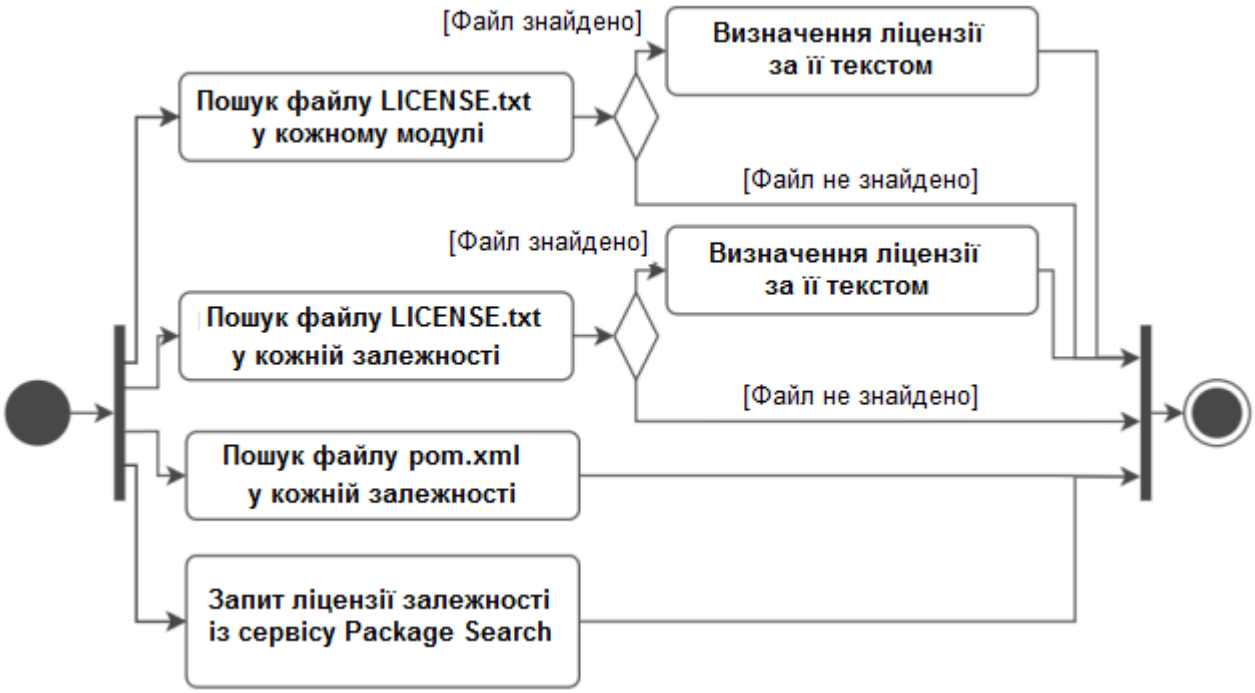

Рисунок 2.4 – Процес вилучення інформації про ліцензії компонентів проєкту

Вилучення інформації про ліцензії модулів проєкту. Цей процес відбувається в такий спосіб. З IntelliJ Platform витягується інформація про всі модулі проєкту. Далі проглядаються кореневі директорії кожного модуля на наявність файлу під назвою LICENSE.txt або за подібними йому. Знайдені файли повинні містити повний текст ліцензії, який використовується для визначення типу ліцензії. Процес визначення типу ліцензії за її повним текстом описано далі. В результаті ми отримуємо список модулів проєкту зі знайденими ліцензіями.

Необхідно відзначити, що використання IntelliJ Platform для отримання даних про модулі проєкту дозволяє відмовитися від визначення ліцензій для кожної директорії проєкту та визначати ліцензії лише для модулів. Для цього необхідно переглядати лише кореневі директорії кожного модуля у пошуках ліцензії. У такий спосіб вдалося зробити процес визначення ліцензій більш оптимальним, ніж перегляд усіх директорій проєкту.

Крім цього IntelliJ Platform використовується для отримання інформації про бібліотеки модуля, що дозволяє врахувати абсолютно всі бібліотеки модуля при визначенні допустимих ліцензій та пошуку потенційних порушень ліцензування.

Без використання платформи отримання інформації про всі бібліотеки є нетривіальним завданням. Наприклад, якщо проєкт використовує систему монтування Gradle, то аналіз Gradle скрипта не дозволить виявити всі бібліотеки, що підключаються за допомогою Gradle плагінів або завдань Gradle. IntelliJ Platform позбавлена таких недоліків. Тому її використання дає плагіну перевагу в порівнянні з іншими інструментами , підвищує точність визначення потенційних порушень чи цензування і коректність рекомендацій ліцензій.

Вилучення ліцензій бібліотек. Після вилучення ліцензій з модулів відбувається вилучення ліцензій бібліотек проєкту, які зберігаються у вигляді jar -архівів. У такому архіві містяться всі необхідні файли для роботи бібліотеки . Серед цих файлів може бути файл LICENSE.txt з повним текстом ліцензії бібліотеки, за яким можна визначити тип ліцензії. Крім цього в jar -архіві повинен бути файл pom.xml, який містить у собі метаінформацію про бібліотеку, включаючи назву її ліцензії. Якщо такий файл в архіві є, то вилучення інформації про ліцензію залежності зводиться до визначення ліцензії за її назвою. Це не завжди просто зробити. На даний момент немає строго формалізованих назв ліцензій, тому назва однієї і тієї ж ліцензії може різнитися в метаданих різних бібліотек. Для розпізнавання ліцензії за назвою в плагіні використовуються регулярні висловлювання.

Однак на практиці досить часто виявляється так, що бібліотеки постачаються без файлу з метаінформацією або в такому файлі відсутня інформація про ліцензію залежності. У такому разі дізнатися про ліцензію бібліотеки без запитів до сторонніх ресурсів практично неможливо.

На даний момент існує кілька репозиторіїв залежностей і сервісів для хостингів IT -проєктів, які можуть містити інформацію про ліцензію конкретної бібліотеки. Реалізація вилучення інформації про ліцензії бібліотек з усіх цих джерел складна і вимагає багато ресурсів на розробку та налагодження.

Альтернативним рішенням є використання сервісу для пошуку Java бібліотек Package Search [28] для отримання інформації про ліцензії бібліотек. Сервіс індексує популярні репозиторії та хостинги Java бібліотек та надає API для

отримання повної інформації про конкретну залежність, включаючи ліцензію бібліотеки та, наприклад, посилання на репозиторій на GitHub. Package Search є продуктом компанії JetBrains, тому не є стороннім рішенням щодо плагіна, що розробляється.

Для визначення ліцензії бібліотеки з допомогою сервісу Package Search було реалізовано компонент Package Search Client, який відправляє запити сервісу і перетворює отриману інформацію у внутрішнє подання плагіна.

Незважаючи на всі зручності роботи з Package Search, у нього є недоліки . Перший недолік – сервіс не індексує репозиторій bintray/jcenter [29], в якому опубліковано безліч Java -бібліотек, яких немає в інших репозиторіях. Цей недолік перестав бути актуальним у зв'язку з новиною про швидке закриття репозиторію bintray /jcenter [30]. Усі бібліотеки, які в ньому знаходилися, повинні незабаром мігрувати в інші репозиторії, наприклад Maven Central [31] або на добре відомий GitHub.

Інший недолік полягає в тому, що Package Search надає інформацію про ліцензію бібліотеки в цілому, а не для кожної конкретної версії. Якщо різні версії бібліотеки знаходяться під різними ліцензіями, то сервіс повертає список ліцензій без уточнення , яка ліцензія якій версії відповідає.

Також необхідно зазначити, що алгоритм контролю ліцензій при пошуку потенційних порушень враховує всі надані сервером ліцензії. Це дозволяє уникнути деяких ситуацій із неправильним визначенням ліцензії бібліотеки, які можуть призводити до незнайдених реальних порушень ліцензій. Разом з тим, використання всіх ліцензій збільшує кількість хибних потенційних порушень, які знаходить плагін. Однак у даному випадку не пропустити реальне порушення важливіше, ніж потенційно знизити кількість хибнопозитивних спрацьовувань.

Визначення ліцензії на повний текст. Окремо необхідно розглянути процес визначення ліцензії на її повний текст.

Цей процес представлений на рисунку 2.5.

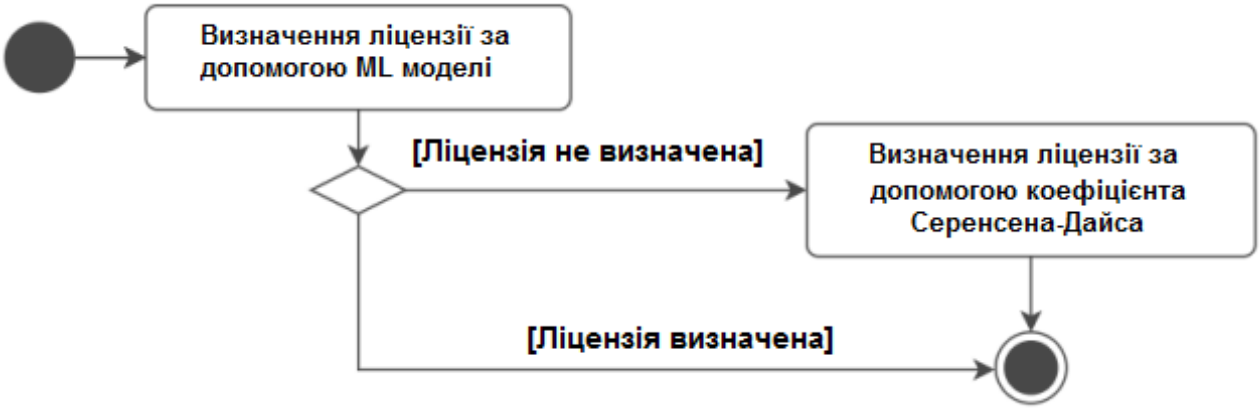

Рисунок 2.5 – Процес визначення ліцензії за її повним текстом

Для визначення ліцензії за її текстом використовують два детектори. Перший детектор (ML) використовує модель машинного навчання, Модель написана мовою Python за допомогою бібліотеки CatBoost [32]. Для використання моделі в Kotlin плагіні, готова модель була експортована у форматі ONNX [33] та інтегрована в плагін за допомогою бібліотеки KInference [34] від компанії JetBrains. Модель підтримує роботу із 12 ліцензіями. Дане обмеження випливає з навчального набору даних, який був розміщений лише для 12 ліцензій.

Для визначення інших ліцензій та випадків, коли ML детектор не зміг визначити тип ліцензії, використовується детектор на основі коефіцієнта подібності Серенсена-Дайса (DSC) [35]. Коефіцієнт добування для двох наборів елементів X і Y визначається як:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|},$$

де $|X|$ і $|Y|$ — відповідні потужності наборів_. Чим ближче значення коефіцієнта до одиниці, тим більше набори елементів схожі.

Детектор на основі коефіцієнта подібності працює наступним чином . Кожна підтримувана ліцензія містить заздалегідь підготовлений набір слів оригінального тексту. Детектор приймає на вхід текст невідомої ліцензії та розбиває його на слова.

Отриманий набір слів порівнюється за допомогою коефіцієнта Серенсена-Дайса з наборами слів оригінальних ліцензій. Збіг зараховується, коли коефіцієнт подібності більший за порогове значення, яке дорівнює 0.95. Таким чином, детектор вимагає збігу з оригінальним текстом ліцензії на 95% і допускає невеликі зміни тексту.

Даний коефіцієнт подібності та значення порога подібності обрано не випадково. Коефіцієнт Серенсен-Дайса з порогом 0.95 лежить в основі інструмента для визначення ліцензій licensee [20], який використовується для визначення ліцензій проєктів на популярному хостингу GitHub. Тому цей метод визначення ліцензій є перевіреним рішенням і показує добрий результат.

Обидва детектори ліцензій містяться в компоненті License Detectors модуля Core. Завдяки їм розроблений плагін підтримує роботу з 16 ліцензіями.

У таблиці 2.1 представлені підтримувані ліцензії та методи їх визначення у плагіні.

Таблиця 2.1 – Підтримувані ліцензії та методи їх визначення у плагіні

Ліцензія	**Метод визначення**
1	**2**
GNU Affero General Public License v3.0 only	ML+DSC
Apache License 2.0	ML+DSC
BSD 2-Clause “Simplified” License	ML+DSC
BSD 3-Clause “New” or “Revised” License	ML+DSC
Common Development and Distribution License 1.0	DSC
Eclipse Public License 1.0	DSC
GNU General Public License v2.0 only	ML+DSC
GNU General Public License v2.0 w/Classpath exception	DSC
GNU General Public License v3.0 only	ML+DSC
ISC License	ML+DSC

Продовження таблиці 2.1

1	2
GNU Lesser General Public License v2.1 only	ML+DSC
GNU Lesser General Public License v3.0 only	ML+DSC
MIT License	ML+DSC
Mozilla Public License 1.1	DSC
Mozilla Public License 2.0	ML+DSC
Do What The F *** You Want To Public License	ML+DSC

Дані ліцензії покривають приблизно 95% всіх Java -проєктів. Ці ліцензії були обрані на основі результатів дослідження [1], в якому міститься інформація про найпопулярніші ліцензії Java - проєктів на GitHub. Незважаючи на те, що в дослідженні розглядаються популярні ліцензії сімейства GPL і LGPL з префіксом or-later, у плагіні підтримані суворіші версії цих ліцензій з префіксом only. Ліцензії з префіксом or-later, на відміну від ліцензій з префіксом only, дозволяють використовувати ліцензоване ПЗ під тією ж ліцензією більш старшої версії. Наприклад, бібліотеку з ліцензією GPL-2.0-or-later можна використовувати при розробці продукту з ліцензією GPL-3.0-only, а ліцензія GPL-2.0-only не надає такої можливості.

Тексти версій only і or-later дуже схожі і можуть відрізнятися лиш одним словосполученням, що погано позначається на точності детектування. Невеликі відмінності в текстах таких ліцензій спричиняють різні правила сумісності з старішими версіями ліцензій. Тому для уникнення некоректної роботи з ліцензіями сімейств GPL і LGPL плагін підтримує роботу з суворішими версіями цих ліцензій.

2.2.2 Визначення допустимих ліцензій для модуля проєкту

На основі інформації про ліцензії компонентів, отриманої на попередньому етапі, відбувається процес визначення допустимих ліцензій для кожного модуля та пошук потенційних порушень ліцензування .

На даному етапі для кожного модуля визначено ліцензію самого модуля, а також ліцензії його підмодулів та бібліотек. Якщо у модуля немає ліцензії, то використовується ліцензія модуля батька, якщо він існує. Також, якщо підмодуль не має ліцензії, то всі його підмодулі та бібліотеки враховуються при обробці модуля батька.

Процес визначення допустимих ліцензій продемонстрований на рисунку 2.6.

Набір допустимих ліцензій модулів визначається наступним чином . Для кожної підтриманої в плагіні ліцензії задані два набори сумісних з нею ліцензій. Один із них призначений для ліцензій бібліотек модуля. Він містить ліцензії модуля, які не порушуватимуть цю ліцензію, якщо вона є ліцензією бібліотеки.

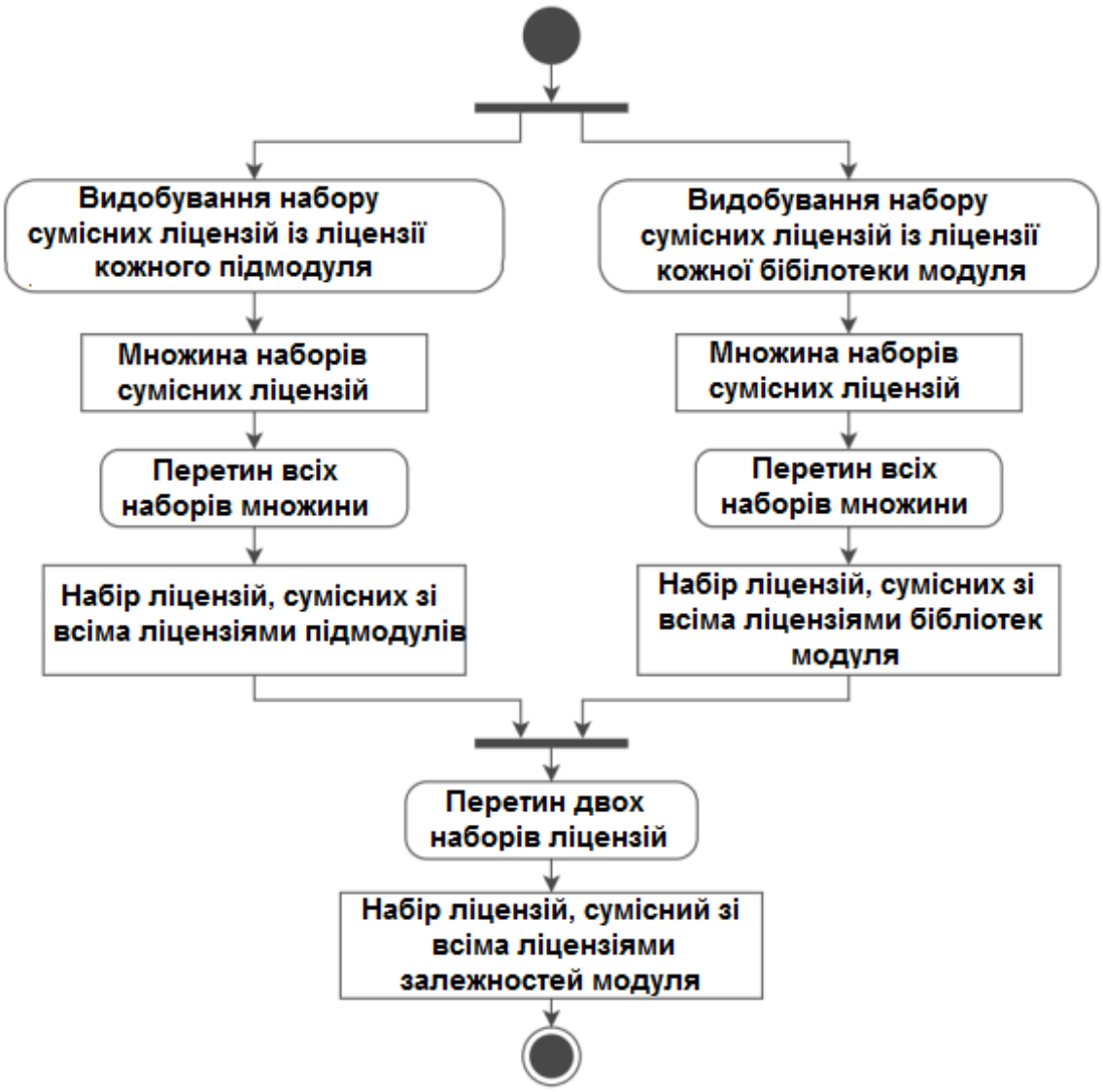

Рисунок 2.6 – Визначення допустимих ліцензій модуля

Такий набір витягується з кожної ліцензії бібліотек модуля . В результаті одержуємо множину наборів допустимих ліцензій. Перетинаючи всі набори сумісних ліцензій, отримуємо один набір з ліцензій, які не порушують жодної ліцензії бібліотеки цього модуля.

Подібна процедура відбувається для ліцензій підмодулів. Для цього використовується інший набір сумісних ліцензій. Він містить допустимі ліцензії модуля, які не порушуватимуть дану ліцензію, якщо вона є ліцензією підмодуля. Такий набір видобувається з кожної ліцензії підмодулів. В результаті отримуємо множину наборів сумісних ліцензій. Перетинаючи всі набори сумісних ліцензій отримуємо один набір з ліцензій, які не порушують жодної ліцензії підмодуля цього модуля.

Отримані два набори ліцензій знову перетинаються. В результаті отримуємо набір із допустимих ліцензій модуля, які не порушують ні ліцензії бібліотек, ні ліцензії підмодулів. Знайдені допустимі ліцензії модуля дозволяють надати рекомендацію щодо вибору ліцензії та допомогти користувачеві вибрати ліцензію, не замислюючись про порушення ліцензування.

2.2.3 Пошук потенційних порушень

Окремо необхідно розглянути пошуки потенційних порушень ліцензування. Процес пошуку наведено на рисунку 2.7.

Потенційні порушення ліцензій визначаються як між ліцензією модуля та ліцензією бібліотеки, так і між ліцензією модуля та ліцензією підмодуля. Процес пошуку потенційних порушень багато в чому схожий на визначення допустимих ліцензій модуля. Для визначення порушення між ліцензією модуля та ліцензією бібліотеки з ліцензії бібліотеки виймається набір допустимих ліцензій модуля. Цей набір містить усі ліцензії, які може мати модуль і які не порушуватимуть ліцензію бібліотеки.

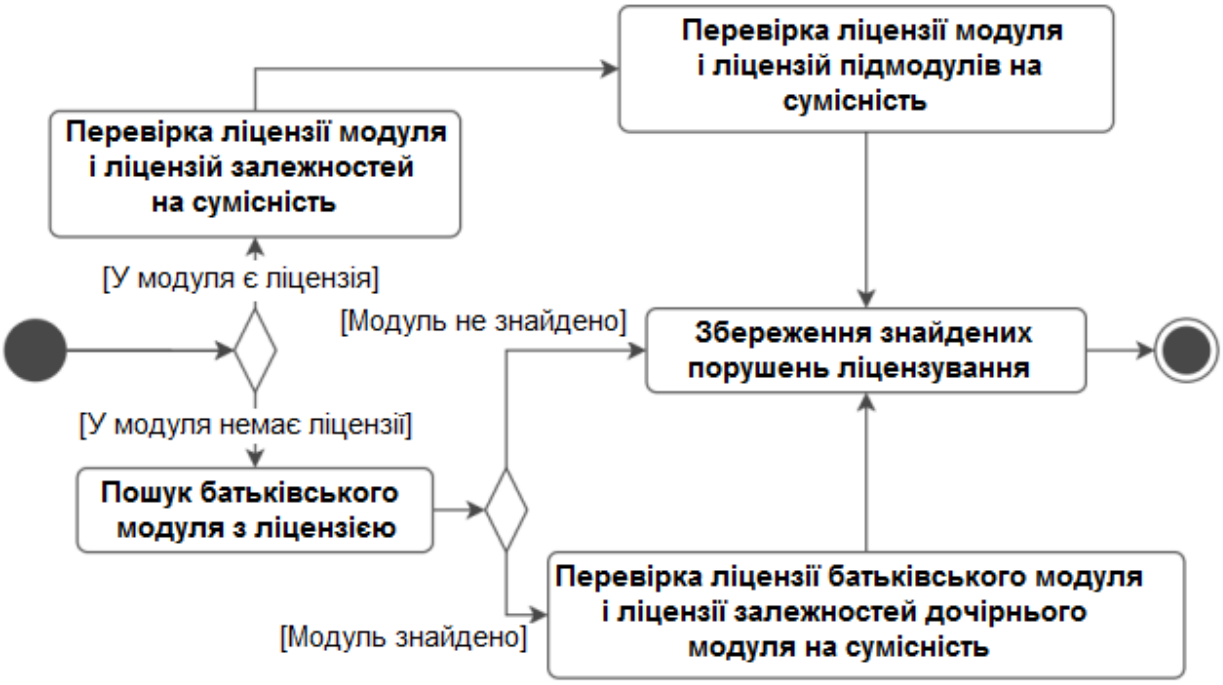

Рисунок 2.7 – Процес пошуку потенційних порушень ліцензування

Якщо поточна ліцензія модуля входить до цього набору, порушення ліцензії немає. Якщо поточна ліцензія до набору не входить, то плагін реєструє потенційне порушення ліцензії та виводить розробнику відповідну інформацію. Подібна процедура проводиться для кожної бібліотеки кожного модуля. Таким чином, плагін визначає всі потенційні порушення пов'язані з порушенням ліцензій бібліотек.

Розглянемо приклад визначення порушення ліцензії бібліотеки. Приклад представлений на рисунку 2.8.

Нехай модуль з ліцензією Apache-2.0 використовує бібліотеку з цензією GPL-2.0-only. З ліцензії GPL-2.0 тільки виймається набір можливих ліцензій модуля, які не порушують ліцензію бібліотеки. Плагін виявляє, що ліцензія Apache-2.0 не входить до цього набору ліцензій та реєструє відповідне порушення ліцензії.

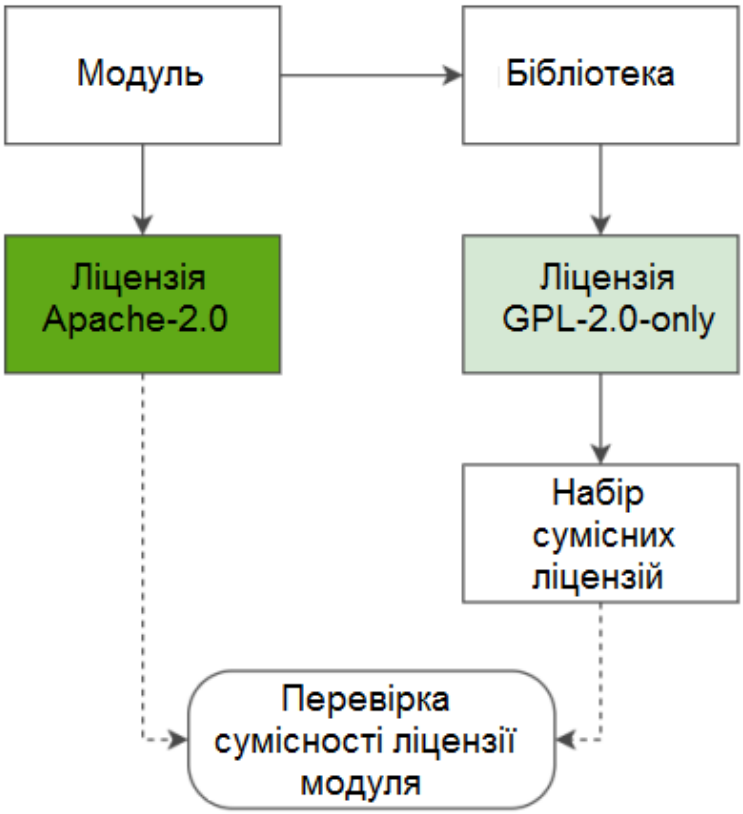

Рисунок 2.8 – Приклад визначення порушення ліцензії бібліотеки

Аналогічним чином відбувається визначення потенційного порушення між ліцензій модуля та ліцензією підмодуля. Для цього з ліцензії підмодуля виходить набір допустимих ліцензій. Цей набір містить усі ліцензії, які може мати модуль та які не порушуватимуть ліцензію підмодуля. Якщо поточна ліцензія модуль входить у цей набір, то порушення немає. Якщо поточна ліцензія в набір не входить, то плагін реєструє чи потенційне порушення ліцензії і виводить розробнику відповідну інформацію. Ця процедура проводиться для кожного підмодуля кожного модуля. В результаті плагін визначає всі потенційні порушення, пов'язані з порушенням ліцензій підмодулів.

Розглянемо приклад визначення порушення ліцензії підмодуля. Приклад наведений на рисунку 2.9.

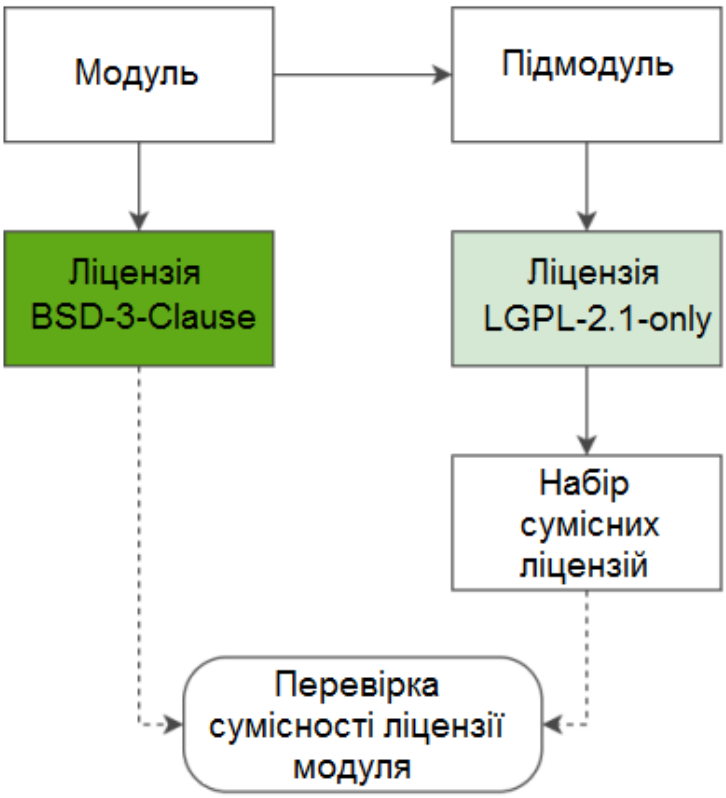

Рисунок 2.9 – Приклад визначення порушення ліцензії підмодуля

Нехай модуль з ліцензією BSD-3-Clause має підмодуль з ліцензією LGPL-2.1-only. З ліцензії LGPL-2.1-only виймається набір можливих ліцензій модуля, які не порушують ліцензію підмодуля. Плагін виявляє, що ліцензія BSD-3-Clause не входить до цієї ліцензії і реєструє порушення ліцензії підмодуля.

2.3 Висновки до другого розділу

У ході проєктування була розроблена архітектура плагіна. Концептуально плагін містить два модулі : Core та Integration. Мова програмного втілення – Kotlin.

Запропоновано алгоритм контролю ліцензій, покрокове виконання якого забезпечує вилучення та визначення інформації про дозволені ліцензії проєкту.

При роботі плагіна забезпечена можливість отримання дозволу на підставі ліцензійного компонента проєкту визначати дозволені ліцензії для кожного модуля проєкту і надавати програмісту інформацію щодо ймовірних порушень.

3 ІНТЕРФЕЙС КОРИСТУВАЧА ПЛАГІНА. ТЕСТУВАННЯ

Розроблений плагін надає користувачеві графічний інтерфейс для зручності роботи з ліцензіями проєкту. Для реалізації інтерфейсу використано бібліотеку Swing [36], яка є стандартною бібліотекою для розробки графічного інтерфейсу в плагінах до IntelliJ IDEA і входить до IntelliJ Platform SDK.

В рамках тестування плагіна необхідно вивчити потенційні порушення ліцензування, які знаходить плагін, а також проаналізувати найчастіші знайдені порушення між ліцензіями.

3.1 Tool Window

Основним графічним інтерфейсом плагіна є вікно, що згортається – Tool Window. Вікно містить дві вкладки та надає основну інформацію про ліцензії проєкту.

Перша вкладка, Project License, містить інформацію про ліцензію кореневого модуля проєкту з докладним описом дозволів та обмежень ліцензії. Крім того, вкладка містить потенційний список порушень ліцензування в проєкті, які плагін зміг виявити. Project License також надає можливість експортувати список порушень ліцензування у форматі json і перезапустити алгоритм контролю ліцензій проєкту. Ця вкладка представлена на рисунку 3.1

Друга вкладка, Package License, демонструє інформацію про знайдені бібліотеки проєкту та їх ліцензії. Також на цій вкладці є можливість фільтрації залежностей за приналежністю до модуля і пошуковим запитом. Вкладка Package License наведена на рисунку 3.2.

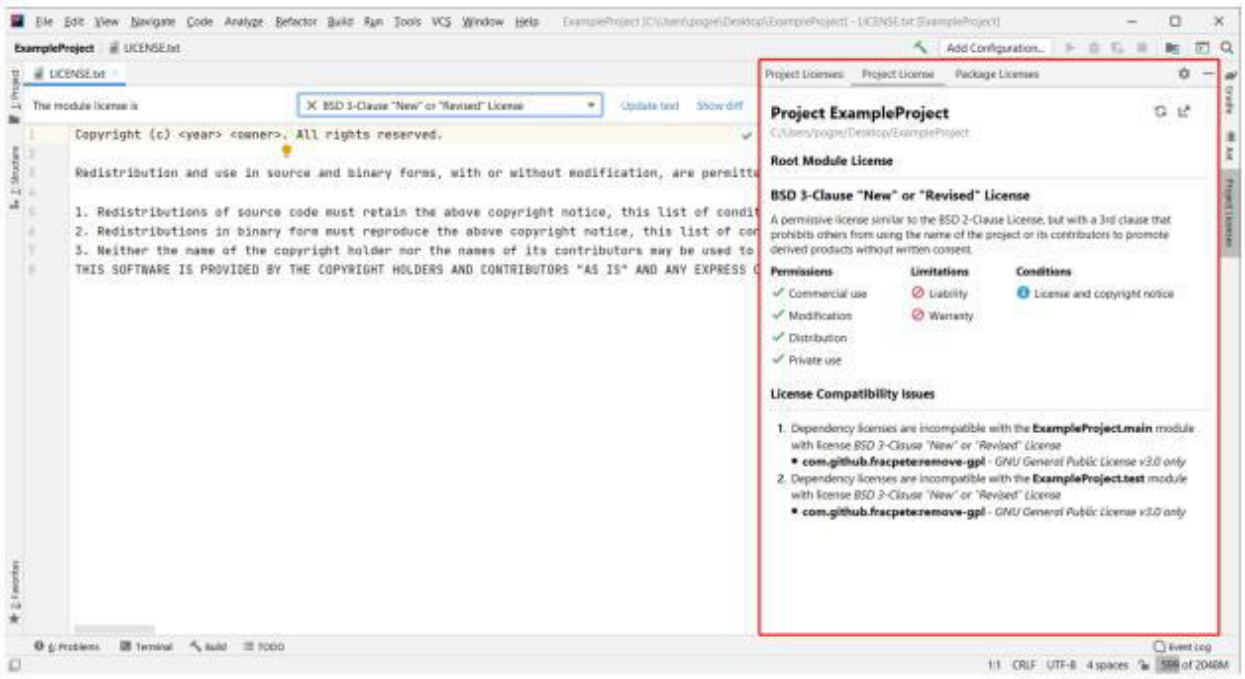

Рисунок 3.1 – Вкладка Project License

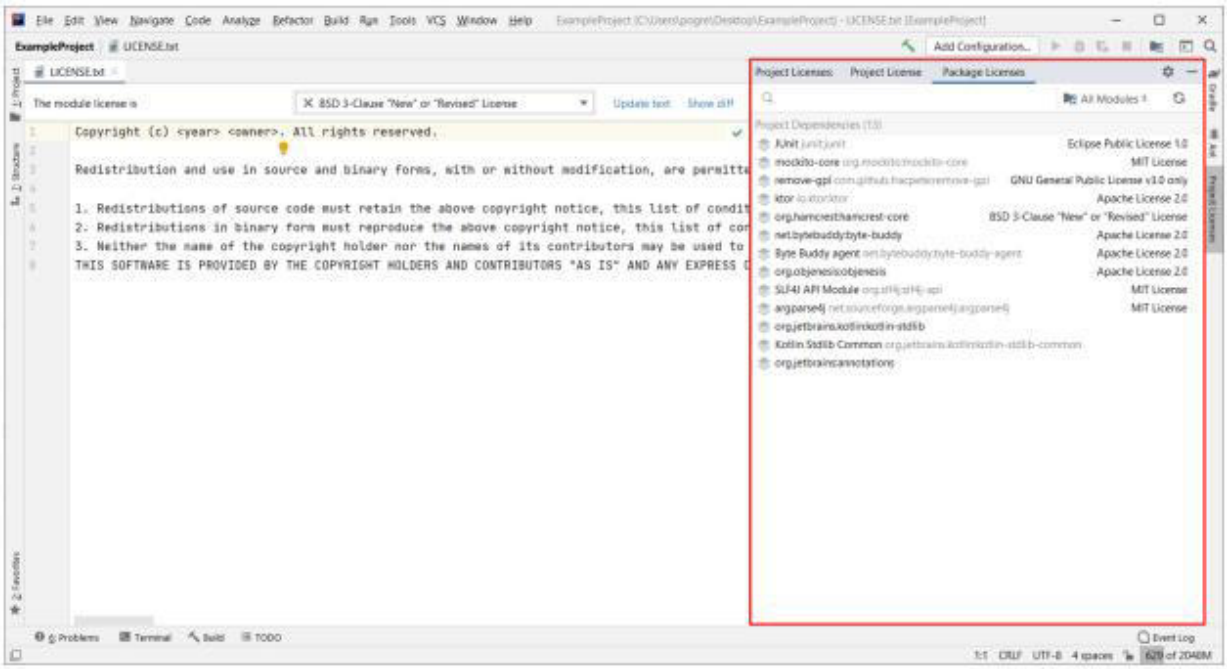

Рисунок 3.2 – Вкладка Package License

3.2 License Editor Notification

Окрім вікна Tool Window, плагін надає панель License Editor Notification. Дана панель знаходиться у верхній частині редактора файлів з ліцензією. Вона

дозволяє подивитися, які з ліцензій сумісна з усіма ліцензіями залежностей модуля або проєкту та вибрати ту, яка найбільш придатна. Крім того, панель надає можливість порівняти поточний текст файлу ліцензії з оригінальним текстом ліцензії. Панель License Editor Notification представлена на рисунку 3.3.

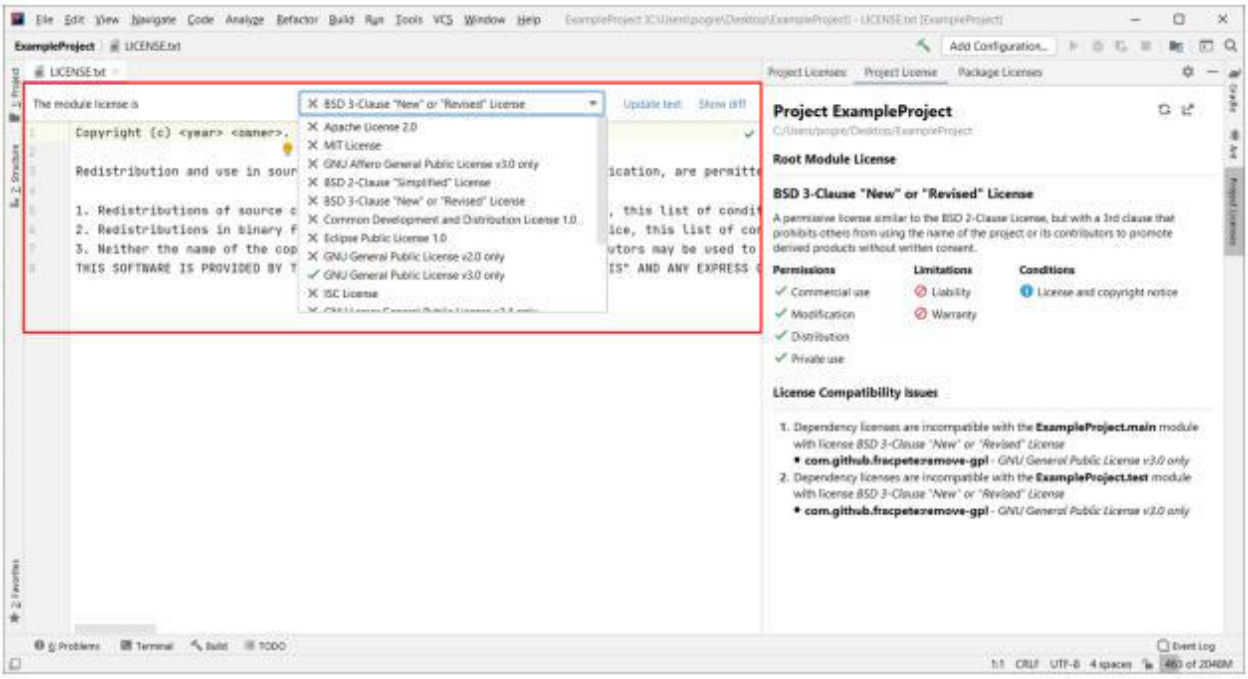

Рисунок 3.3 – Панель редактора для роботи з файлом основної ліцензії проєкту

У цьому прикладі проєкт має залежність з ліцензією GPL 3.0, тому сумісною з нею ліцензією для всього проєкту є лише ліцензія GPL-3.0, що відзначено галочкою. Щоб дізнатися, які порушення виникають під час використання несумісних ліцензій, необхідно вибрати таку ліцензію зі списку ліцензій. Після цього у вікні ToolWIndow з'явиться відповідний перелік потенційних порушень.

Крім цього плагін надає можливість створити файл з найбільш допустимою ліцензією модуля. Завдяки цим можливостям користувач, який практично не має знань про ліцензії, зможе вибрати коректну ліцензію для свого проєкту та не припуститися помилки

3.3 Inlay License Hints

Розроблений плагін також надає підказки з ліцензіями бібліотек під час редагування скрипту системи складання проєкту. Напроти кожної команди з під'єднанням бібліотеки з'являється підказка з назвою ліцензії цієї бібліотеки. Такі підказки дозволяють користувачеві відразу дізнатися ліцензію залежності безпосередньо в редакторі коду.

Дані підказки реалізовані для Maven, Groovy Gradle та Kotlin Gradle скриптів. Приклад роботи підказок у Kotlin Gradle скрипті представлений на рисунку 3.4.

Рисунок 3.4 – Підказки ліцензій бібліотек у Kotlin Gradle скрипті

3.4 Вбудовування у меню New

Також плагін додає новий пункт у меню «Створити новий файл...» IDE під назвою «Файл ліцензії модуля» (рисунок 3.5).

Якщо проєкт не має ліцензії, користувач може скористатися цією функцією, і плагін визначить ліцензії, сумісні з усіма бібліотеками проєкту, і запропонує користувачеві ту, яка є найбільш дозволена.

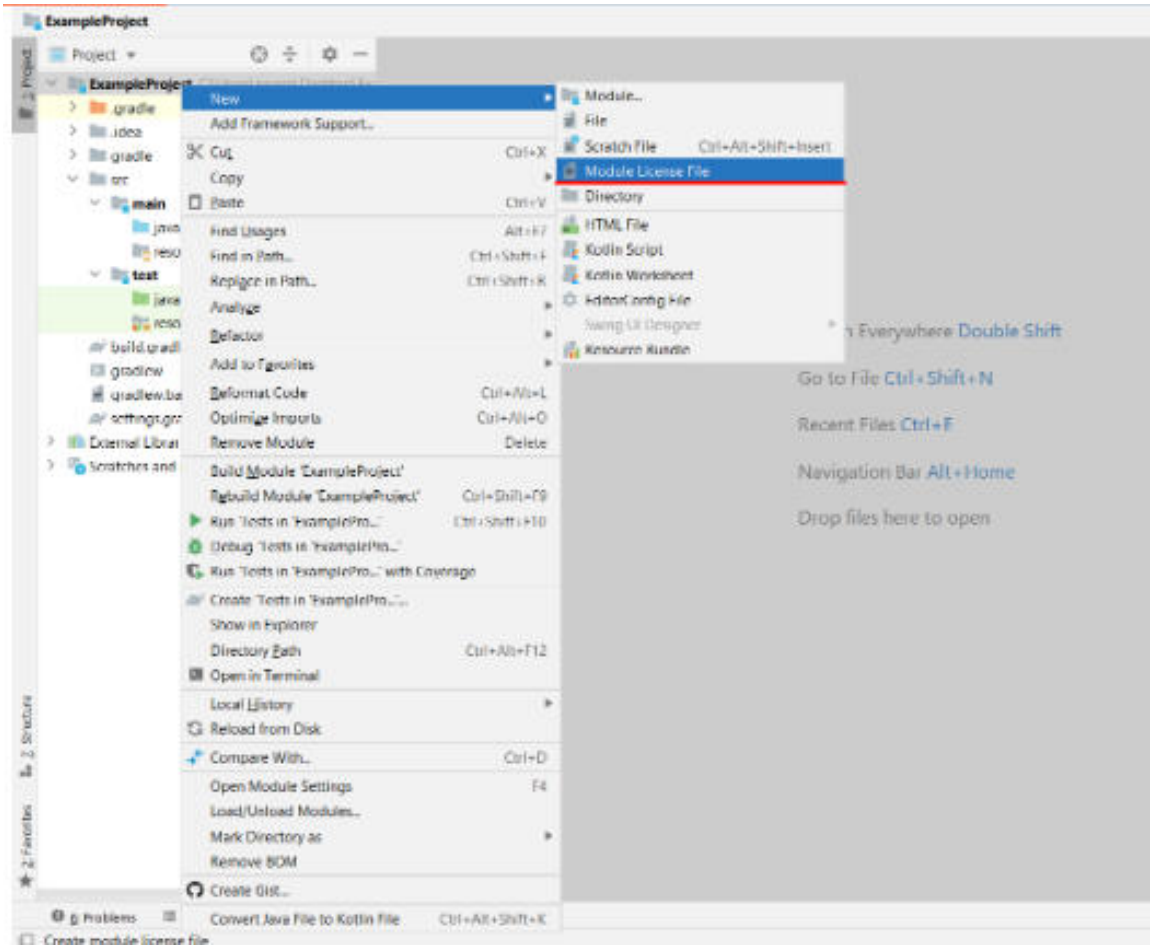

Рисунок 3.5 – Вікно вбудовування у меню New

Таким чином, навіть найменш досвідчений користувач зможе керувати своїми ліцензіями та не робити при цьому помилок. У майбутньому користувач завжди зможе змінити свою ліцензію при допомозі панелі сповіщень редактора ліцензій, котра описана вище.

3.5 Тестування плагіна

Апробація плагіна проводилася на 100 Java проєктах на GitHub з найбільшою кількістю зірок. Для запуску плагіна на множині проєктів був написаний спеціальний скрипт, який запускає IntelliJ IDEA в headless режимі (без графічного інтерфейсу) і збирає інформацію про порушення ліцензій з плагіна. Однак у процесі апробації виявилося, що частина проєктів некоректно імпортуються за допомогою фреймворку для тестування IntelliJ Platform. Тому такі проєкти були оброблені

вручну та відкриті в IDE у стандартному режимі.

За результатами апробації у 100 проєктах плагін виявив 762 потенційні порушення між ліцензією бібліотеки та ліцензією модуля. Десять найпопулярніших пар ліцензій у потенційних порушеннях ліцензування наведені на рисунку 3.6.

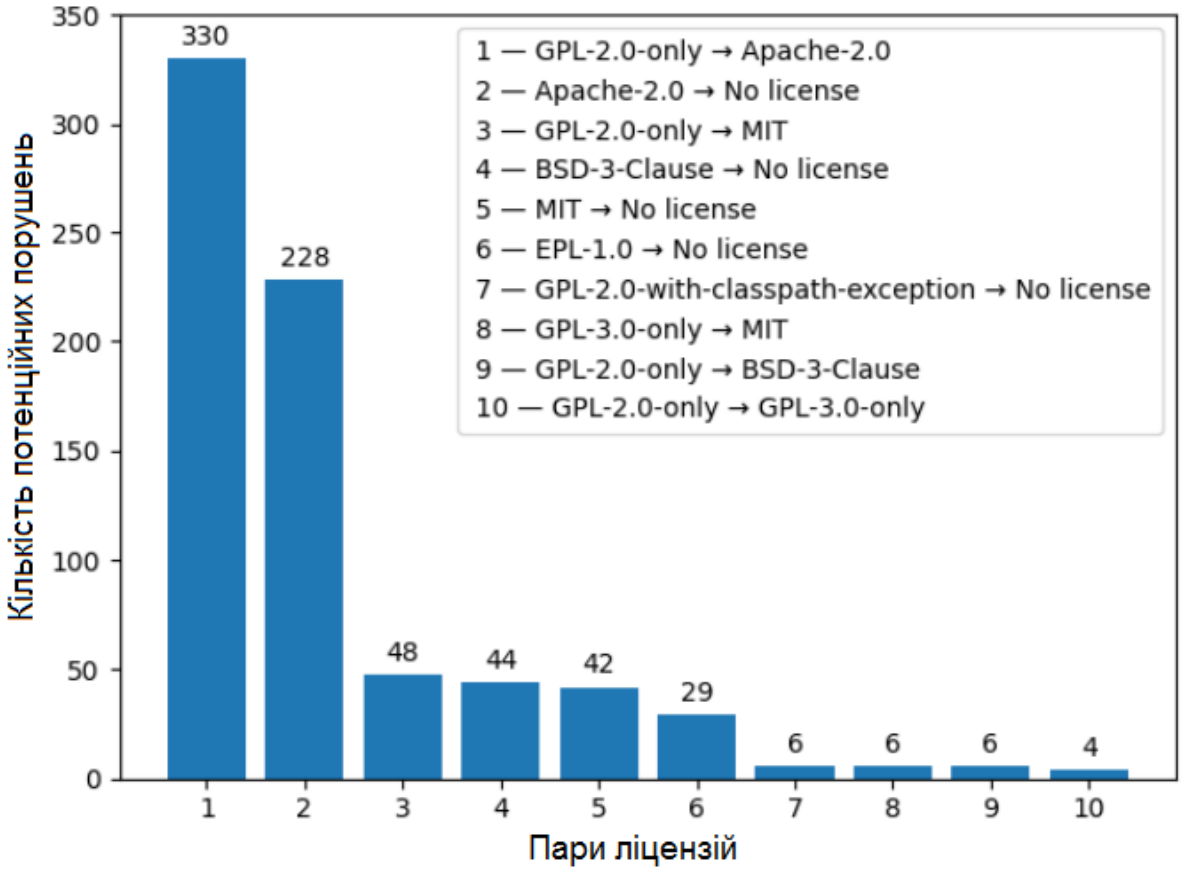

Рисунок 3.6 – Десять найпопулярніших пар ліцензій (бібліотека -> модуль) у потенційних випадках порушення правил ліцензування

Серед знайдених порушень ліцензій виділяється група із 314 випадків використання бібліотек з дозвільними ліцензіями (Apache-2.0, MIT, BSD-3-CLause) у модулях без ліцензії (No license). Дані потенційні порушення пов'язані з нестандартними або модифікованими текстами ліцензій модулів, що не дозволяє програмно визначити тип ліцензії. Тим не менш, дані порушення вказують розробнику на можливі проблеми з ліцензією модуля і можуть мотивувати його поміняти ліцензію на одну із загальноприйнятих .

Крім цього, серед виявлених порушень 384 потенційні порушення пов'язані з використанням бібліотек під ліцензією GPL-2.0-only в модулях з дозвільними ліцензіями (Apache-2.0, MIT і BSD-3-CLause). Причиною цих потенційних порушень могла стати серйозна робота з модифікаціями ліцензії GPL-2.0. Бібліотеки з ліцензією GPL-2.0-with-classpath-exception можуть використовуватися в проєкті з дозвільною ліцензією, наприклад з Apache-2.0, на відміну від бібліотек з ліцензією GPL-2.0-only. Однак тексти ліцензій GPL-2.0- with-dasspath-exception і GPL-2.0-only можуть різнитися лише в одному реченні. У випадках, коли немає можливості визначити точну модифікацію ліцензії, плагін вибирає ту, котра найбільше обмежує, а саме GPL-2.0-only, що призводить до появи потенційних порушень. Такий процес роботи з модифікаціями ліцензій GPL-2.0 дозволяє не пропустити реальні порушення ліцензії GPL-2.0-only, звертає увагу розробника на підозрілі бібліотеки та спонукає перевірити коректність знайдених порушень вручну.

Необхідно відзначити, що деякі знайдені потенційні порушення після ретельної ручної перевірки можуть виявитися помилковими і не будуть порушеннями як такими. Однак одним із завдань плагіна є завдання відсіювання свідомо коректних поєднань ліцензій та надання розробнику інформації про доглядові випадки поєднання ліцензій. Розробник, отримавши інформацію про потенційні порушення, може вручну перевірити їх на коректність або усунути порушення відразу, відмовившись від бібліотеки з несумісною ліцензією або змінивши ліцензію відповідного модуля.

3.5 Висновки до третього розділу

Створений плагін з метою забезпечення зручної роботи користувача володіє графічним інтерфейсом. Розроблено три види закладок (Tool Window, License Editor Notification, Inlay License Hints), в яких відображається вся необхідна інформація щодо ліцензій проєкту. Також передбачена можливість вбудовування плагіна у пункт меню New популярного IDE для забезпечення можливості

керування своїми ліцензіями навіть для недосвідченого користувача.

Необхідно зазначити, що за результатами проведеного тестування плагін не виявив потенційних порушень ліцензій між модулями самого проєкту. Ймовірно, це пов'язано з тим, що розробники зазвичай ліцензують весь проєкт повністю, а не кожен модуль окремо. Тому порушень ліцензування між модулями проєкту не виникає.

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

В цьому розділі необхідно дослідити важливі питання, котрі стосуються безпеки життєдіяльності та основ охорони праці.

4.1 Навчання працюючих і інструктажі з охорони праці

Однією із складових ефективної роботи з профілактики виробничого травматизму є належна підготовка, навчання та підвищення кваліфікації працівників з питань охорони праці. Загальний порядок проведення навчання з питань охорони праці встановлений Законом України «Про охорону праці» (ст. 18. «Навчання з питань охорони праці»).

Виконання вимог Закону України «Про охорону праці» в частині проведення навчання та перевірки знань з питань охорони праці здійснюється відповідно до Типового положення про порядок проведення навчання і перевірки знань з питань охорони праці, затвердженого наказом Держкомітету України з нагляду за охороною праці 26 січня 2005 р. № 15 (далі — Типове положення).

Нагляд за дотриманням вимог Типового положення здійснюють органи державного нагляду за охороною праці, а координацію та методичний супровід — Головний навчально-методичний центр та навчальні підрозділи експертно-технічних центрів Держгірпромнагляду.

Вивчення предмета «Охорона праці» при підготовці, перепідготовці та підвищенні кваліфікації працівників, які залучаються до виконання робіт з підвищеною небезпекою, на підприємстві регламентується п. 2.3 Типового положення. На підприємствах згідно з п. 1.1 Додатку 3 Типового положення навчання та перевірку знань з питань охорони праці повинні проходити керівники, заступники керівників, головні спеціалісти, керівники основних виробничих та технічних служб, безпосередньо пов’язані з організацією безпечного ведення робіт. Крім цього, згідно з п. 5 Додатку 3, навчання та перевірку знань з питань охорони праці мають проходити керівники, спеціалісти служб охорони праці, члени комісій

з перевірки знань з питань охорони праці, особи, відповідальні за технічний стан і безпечну експлуатацію об’єктів підвищеної небезпеки підприємств.

Типове положення встановлює порядок та місце проведення навчання та перевірки знань з питань охорони праці посадових осіб (п. 5.2 та п. 5.3). Посадові особи, перелік яких наведено в п. 5.2, проходять навчання у Головному навчально-методичному центрі Держнаглядохоронпраці. Перевірка знань цієї категорії посадових осіб проводиться комісією, створеною наказом Держгірпромнагляду.

Організацію навчання та перевірки знань з питань охорони праці працівників на підприємстві здійснюють працівники служби кадрів або інші спеціалісти, яким роботодавець доручив організацію цієї роботи. Навчання та перевірка знань з питань охорони праці працівників (виконавців і посадових осіб), які не залучаються до виконання робіт підвищеної небезпеки, проводиться не рідше ніж один раз на три роки. Посадові особи та працівники, які виконують роботи підвищеної небезпеки, проходять спеціальне навчання та перевірку знань відповідних нормативно-правових актів з охорони праці не рідше одного разу на рік.

Посадові особи малих підприємств (п. 5.4), які не мають можливості створити власні комісії з перевірки знань з питань охорони праці та провести навчання з питань охорони праці, проходять навчання та перевірку знань в навчальних закладах, які мають відповідний дозвіл. Спеціальне навчання з питань охорони праці може проводитись безпосередньо на підприємстві або навчальним закладом, який має відповідний дозвіл. При проведенні такого навчання на підприємстві навчальні плани та програми розробляються з урахуванням конкретних видів робіт, виробничих умов і функціональних обов’язків працівників і затверджуються наказом керівника підприємства. Періодичність інструктажів, навчання та перевірки знань з питань охорони праці залежить від видів виконуваних робіт та встановлюється Типовим положенням. Перевірка знань з питань охорони праці після проведення спеціального навчання проводиться комісією підприємства.

Якщо на підприємстві неможливо створити комісію з перевірки знань з питань охорони праці (п. 4.4 Типового положення), перевірка знань проводиться комісією спорідненого підприємства або Теруправління Держгірпромнагляду.

Всі працівники та посадові особи підприємства, включаючи посадових осіб, відповідальних за виконання робіт підвищеної небезпеки (крім зазначених в п. 5.2 та п. 5.3 Типового положення), проходять навчання та перевірку знань з питань охорони праці на підприємстві. Типове положення не зобов'язує, але й не забороняє проводити навчання всіх виконавців та посадових осіб (особливо тих, що виконують роботи підвищеної небезпеки) в навчальних закладах. У нашій країні є багато підприємств, де таке навчання проводиться, і це має позитивні наслідки. Ті витрати, які при цьому несуть підприємства, перекриваються створенням більш безпечних умов праці і в результаті збереженням життя та здоров'я працівників.

Також в навчальних закладах проходять навчання та перевірку знань із загальних питань охорони праці всі посадові особи та фахівці, які проводять інструктажі або навчання підлеглих працівників з питань охорони праці, виконують роботи з проектування об'єктів, а також інші працівники, незалежно від того, передбачено таке навчання Типовим положенням чи ні.

4.2 Санітарно-гігієнічні вимоги до умов праці

На робочих місцях працівників, які відповідальні за експлуатацію сервісу управління механізмом авторських прав на мультимедійні файли, необхідно забезпечити дотримання вимог, затверджених Наказом Мінсоцполітики від 14.02.2018 за № 207 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями».

Приміщення для роботи з ЕОМ мають бути обладнані системами опалення, кондиціонування повітря, або припливно-витяжною вентиляцією. У приміщеннях на робочих місцях мають забезпечуватись оптимальні значення параметрів мікроклімату: температури, відносної вологості та рухливості повітря відповідно до норм та правил, а також ДБН В.2.5-67:2013 «Опалення, вентиляція та кондиціонування», затверджених наказом Мінрегіону від 25.01.2013 р. № 24. Відповідно до санітарних норм мікроклімату виробничих приміщень ДСН

3.3.6.042-99 в офісних приміщеннях, обладнаних ЕОМ, температура повітря повинна становити 22-25°С, відносна вологість повітря – 40-60 %, швидкість руху повітря – не більше 0,1 м/с [42].

Приміщення, призначені для роботи з ЕОМ, повинні мати природне освітлення. У виробничих приміщеннях, обладнаних ЕОМ, необхідно створити належне освітлення. При експлуатації сервісу управління механізмом авторських прав на мультимедійні файли, важливим, з точки зору охорони праці, є забезпечення достатньої величини природного та штучного освітлення, які визначені у НПАОП 0.00-7.15-18 [43]. Природне світло повинно бути бічним, зорієнтованим, як правило, на північ чи північний схід, і забезпечувати коефіцієнт природної освітленості не нижче 1,5%. При виробничій потребі дозволяється експлуатувати ЕОМ у приміщеннях без природного освітлення за узгодженням з органами Держпромгірнагляду та органами й установами санітарно-епідеміологічної служби. Вікна приміщень повинні мати регулювальні пристрої для відчинення, а також жалюзі, штори тощо. Штучне освітлення приміщення з робочими місцями, обладнаними відеотерміналами ЕОМ загального та персонального користування, має бути всеосяжним і рівномірним. У випадку, коли переважають роботи з документами, допускається комбіноване освітлення (додатково до загального освітлення встановлюється світильники місцевого освітлення). Світильники розміщуються збоку від робочих місць (переважно ліворуч), або локально над робочим місцем (при розташуванні відеотерміналів ЕОМ за периметром приміщення). Як джерело світла при штучному освітленні застосовуються, як правило, люмінесцентні лампи. У світильниках місцевого освітлення допускається застосування ламп розжарювання. Рівень освітленості на робочому місці має становити 300-500 лк. При використанні комбінованого освітлення не допускається відблисків на поверхні екрана та збільшення освітлення екрана вище 300 лк.

Орієнтація вікон повинна бути на північ або північний схід, вікна повинні мати жалюзі, які можна регулювати, або штори; кабінети, обладнані комп'ютерною технікою, в навчальних закладах повинні розміщуватись в окремих приміщеннях з

природним освітленням та організованим обміном повітря; стіни, стеля і підлога та обладнання кабінетів комп'ютерної техніки повинні мати покриття із матеріалів з матовою фактурою з коефіцієнтом відбиття: стін — 40- 50 %, стелі — 70 - 80 %, підлоги — 20-30 %, предметів обладнання — 40-60 % (робочого столу — 40-50 %, корпуса дисплею та клавіатури — 30-50 %, стелажів — 40-60 %); поверхня підлоги повинна мати антистатичне покриття та бути зручною для вологого прибирання; забороняється використовувати для оздоблення інтер'єру приміщень комп'ютерних кабінетів полімерні матеріали (дерев'яно-стружкові плити, шпалери, що придатні для миття, плівкові та рулонні синтетичні матеріали та ін), що виділяють у повітря шкідливі хімічні речовини, які перевищують гранично допустимі концентрації; вміст шкідливих хімічних речовин в повітрі дошкільних та учбових приміщень з комп'ютерною технікою не повинен перевищувати середньодобові концентрації [43].

Організація робочого місця фахівця із експлуатації сервісу повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ДСТУ 8604:2015 «Дизайн і ергономіка. Робоче місце для виконання робіт у положенні сидячи. Загальні ергономічні вимоги».

Відстань від екрана до ока фахівців, які працюють за комп’ютером визначається згідно з вимогами ДСанПіН 3.3.2.007-98.

Рівень шуму не повинний перевищувати: на місцях, де працюють програмісти та оператори ЕОМ, 55 дБА, у лабораторіях, де складаються алгоритми та ведеться робота з документацією – 60 дБА, у машинному залі – 65 дБА, у приміщеннях, де розміщені гучні агрегати обчислювальних машин – 75 дБА

4.3. Висновки до четвертого розділу

В цьому розділі проаналізовано важливі питання охорони праці та безпеки в надзвичайних ситуаціях, зокрема висвітлено питання навчання працюючих і інструктажі з охорони праці та дотримання санітарно-гігієнічних вимог до умов праці.

ВИСНОВКИ

У ході цієї роботи було отримано такі результати.

– зроблено огляд предметної області. Розглянуто різні типи відкритих ліцензій та правила сумісності між ними. Також зроблено огляд відомих досліджень про порушення ліцензування під час копіювання коду відкритих проєктів;

– проведено аналіз популярних інструментів для роботи з ліцензіями проєктів, який показав, що більшість інструментів не опрацьовують ліцензії залежностей проєкту. Безкоштовні інструменти не надають користувачеві рекомендації щодо вибору основної ліцензії проєкту і не повідомляють про несумісні ліцензії. Крім того, жоден із інструментів не вбудований в жодне з популярних IDE;

– спроєктовано архітектуру плагіна. Вона дозволила відокремити логіку роботи з ліцензіями від інтерфейсу користувача і спростити підтримку нових детекторів і ліцензій у плагіні;

– реалізовано алгоритм контролю ліцензій проєкту. Реалізовано механізм вилучення інформації про ліцензії модулів та залежностей проєкту, а також реалізовано механізм визначення сумісних ліцензій для кожного модуля проєкту на підставі ліцензій залежностей та підмодулів;

– реалізовано інтерфейс плагіна, який містить докладну інформацію про ліцензії модулів та залежностей, а також список потенційних порушень ліцензування. Крім того, інтерфейс містить рекомендації щодо вибору ліцензії, сумісної з ліцензіями всіх модулів та залежностей проєкту;

– проведене тестування розробленого плагіна на 100 найпопулярніших проєктах на GitHub, що містять код на Java. Плагін виявив 762 потенційних порушень ліцензування, з яких 384 потенційне порушення пов'язане з використанням бібліотек з модифікаціями ліцензії GPL-2.0 у модулях з дозвільною ліцензією.

В результаті цієї роботи була реалізована і випущена пілотна версія плагіна

для роботи з ліцензіями Java-проєктів, який може успішно застосовуватися для пошуку потенційних порушень чи ліцензування.

Плагін може бути покращений головним чином за рахунок розширення списку підтримуваних ліцензій і більш точних інструментів для визначення типу ліцензії за її повним текстом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. A Study of Potential Code Borrowing and License Violations in Java Projects on GitHub / Yaroslav Golubev, Maria Eliseeva, Nikita Povarov, Timofey Bryksin // Proceedings of the 17th International Conference on Mining Software Repositories. New York, NY, USA : Association for Computing Machinery, 2020. P. 54–64.

2. Markovtsev Vadim, Long Waren. Public git archive // Proceedings of the 15th International Conference on Mining Software Repositories - MSR ’18. 2018.

3. SPDX License List. URL: https://spdx.org/ licenses/ (дата звертання: 13.05.2026).

4. von Krogh G., Spaeth S., Haefliger S. Knowledge Reuse in Open Source Software: An Exploratory Study of 15 Open Source Projects // Proceedings of the 38th Annual Hawaii International Conference on System Sciences. 2005. P. 1982–1988.

5. Do Software Developers Understand Open Source Licenses? / Daniel A. Almeida, Gail C. Murphy, Greg Wilson, Mike Hoye // Proceedings of the 25th International Conference on Program Comprehension. ICPC ’17. Buenos Aires, Argentina : IEEE Press, 2017. P. 1–11.

6. Gaudeul Alex. Do Open Source Developers Respond to Competition? The LATEX Case Study // Review of Network Economics. 01 Jun. 2007. Vol. 6, no. 2.

7. Brasseur V.M. Forge Your Future with Open Source: Build Your Skills, Build Your Network, Build the Future of Technology. Pragmatic programmers. Pragmatic Bookshelf, 2018.

8. Growing the impact of Open Source around the world. URL: https:// opensource.org (дата звертання: 13.05.2026).

9. Wikipedia contributors. Software license. URL: https://en.wikipedia.org/w/index.php?title=Software_ license&oldid=991148912 (дата звертання: 14.05.2026).

10. SPDX License List. URL: https://spdx.org/ licenses/ (дата звертання: 15.05.2026).

11. Balter Ben. Open source license usage on GitHub.com. – URL: https://github.blog/ 2015-03-09-open-source-license-usage-on-github-com/ (дата звертання: 16.05.2026).

12. German Daniel M., Di Penta Massimiliano, Davies Julius. Understanding and Auditing the Licensing of Open Source Software Distributions // Proceedings of the 2010 IEEE 18th International Conference on Program Comprehension. ICPC ’10. USA : IEEE Computer Society, 2010. P. 84–93.

13. Sourcerer’s Apprentice and the study of code snippet migration / S. Romansky, C. Chen, B. Malhotra, A. Hindle // ArXiv. 2018. Vol. abs/1808.00106.

14. An Empirical Study of License Violations in Open Source Projects / Arunesh Mathur, Harshal Choudhary, Priyank Vashist et al. // Proceedings of the 2012 35th Annual IEEE Software Engineering Workshop. SEW ’12. USA : IEEE Computer Society, 2012. P. 168– 176.

15. Zhang Hongyu, Shi Bei, Zhang Lu. Automatic Checking of License Compliance // Proceedings of the 2010 IEEE International Conference on Software Maintenance. ICSM ’10. USA : IEEE Computer Society, 2010. P. 1–3.

16. askalono GitHub repository. URL: https:// github.com/jpeddicord/askalono (дата звертання: 17.05.2026).

17. FOSSology GitHub repository. URL: https:// github.com/fossology/fossology (дата звертання: 18.05.2026)

18. go-license-detector GitHub repository. URL: https://github.com/src-d/go-license-detector (дата звертання: 18.05.2026).

19. licensechecker GitHub repository. URL: https: //github.com/boyter/lc (дата звертання: 18.05.2026).

20. licensee GitHub repository. URL: https:// github.com/license/licensee (дата звертання: 19.05.2026).

21. scancode-toolkit GitHub repository. URL: https://github.com/nexB/scancode-toolkit (дата звертання: 19.05.2026).

22. FOSSA official website. URL: https://fossa. com (дата звертання: 20.05.2026).

23. license-maven-plugin GitHub repository. URL тупа: https://github.com/mycila/license-maven-plugin (дата звертання: 20.05.2026).

24. Gradle License Report GitHub repository. URL: https://github.com/jk1/Gradle-License-Report (дата звертання: 20.05.2026)

25. Gradle License Report GitHub repository. URL: https://github.com/jk1/Gradle-License-Report (дата звертання: 20.05.2026).

26. Intellij Platform SDK Docs. URL: https:// plugins.jetbrains.com/docs/intellij/welcome.html (дата звертання: 20.05.2026).

27. Intellij Platform GitHub repository. URL: https://github.com/JetBrains/intellij-community (дата звертання: 20.05.2026).

28. Package Search Official Website. URL: https: //package-search.jetbrains.com/ (дата звертання: 21.05.2026).

29. Bintray JCenter dependency repository. URL: https://bintray.com/bintray/jcenter (дата звертання: 21.05.2026).

30. Sunset Bintray JCenter Official Blog Post. URL: https://jfrog.com/blog/ into-the-sunset-bintray-jcenter-gocenter-and-chartcenter/ (дата звертання 21.05.2026).

31. Maven Central dependency repository. URL: https://mvnrepository.com/repos/central/ (дата звертання: 21.05.2026).

32. CatBoost — gradient boosting on decision trees. URL: https://catboost.ai/ (дата звертання: 21.05.2026).

33. Open Neural Network Exchange. URL: https: //onnx.ai/ (дата звертання: 21.05.2026).

34. KInference. URL: https://github.com/ JetBrains-Research/kinference (дата звертання: 21.05.2026).

35. Dice Lee R. Measures of the Amount of Ecologic Association Between Species. URL: https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.2307/ 1932409 (дата звертання: 21.05.2026).

36. Swing — GUI widget toolkit for Java. URL: https://docs.oracle.com/javase/8/docs/technotes/guides/ swing/index.html (дата звертання: 22.05.2026).

37. Методичні вказівки до виконання кваліфікаційної роботи бакалавра для здобувачів спеціальності 121 – Інженерія програмного забезпечення, всіх форм навчання / укладачі Михалик Д.М., Цуприк Г.Б., Бревус В.М. Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2024. 45 с.

38. Stefanyshyn, I. , Pastukh, O., Stefanyshyn, V. , Baran, I. , Boyko, I. Robustness of AI algorithms for neurocomputer interfaces based on software and hardware technologies CEUR Workshop Proceedings, 2024, 3742, pp. 137–149.

39. Boyko, I. , Petryk, M. , Mudryk, I. , Stoianov, Y., Tsupryk, H. Mathematical Model of the Capacitor Based on Zeolite Material. Proceedings - International Conference on Advanced Computer Information Technologies, ACIT, 2021, pp. 45–48.

40. Tsupryk H. LLM-based Extraction from Resumes / D. Olianin, H. Tsupryk // Advanced Technologies in Scientific Research: collection of scientific papers with proceedings of the 1st International Scientific and Practical Conference, Rotterdam, Netherlands, 20–22 August 2025. – International Scientific Unity, 2025. – 72-76.

41. . R. Petryk, A. Khimich, M. M. Petryk, and J. Fraissard, “Experimental and computer simulation studies of dehydration on microporous adsorbent of natural gas used as motor fuel,” Fuel, Vol. 239, 1324–1330 (2019). https://doi.org/https://doi.org/10.1016/j.fuel.2018.10.134

42. Заікіна Д., Глива В. Основи охорони праці та безпека життєдіяльності. 2019. URL: https://doi.org/10.31435/rsglobal/001 (дата звертання: 10.12.2025).

43. Зеркалов Д.В. Безпека життєдіяльності та основи охорони праці. Навчальний посібник. К.: «Основа». 2016. – 267 с.