Санкт-Петербургский государственный университет

Погребной Дмитрий Андреевич

Выпускная квалификационная работа

Разработка системы проверки совместимости лицензий ПО

Уровень образования: бакалавриат

Направление *09.03.04 «Программная инженерия»*

Основная образовательная программа *СВ.5080.2017 «Программная инженерия»*

Научный руководитель:
доцент кафедры СП, к.т.н. Т. А. Брыксин

Рецензент:
Специалист по машинному обучению
ООО «ИнтеллиДжей Лабс» Е. Г. Булычев

Санкт-Петербург
2021

Saint Petersburg State University

Dmitry Pogrebnoy

Bachelor's Thesis

Development of a software license compatibility check system

Education level: bachelor

Speciality *09.03.04 «Software Engineering»*

Programme *CB.5080.2017 «Software Engineering»*

Scientific supervisor:
Ph.D., Associate professor Timofey Bryksin

Reviewer:
Machine learning engineer, «IntelliJ Labs» Co. Ltd. Egor Bulychev

Saint Petersburg
2021

Оглавление

Введение

На сегодняшний день существует большое количество проектов с открытым исходным кодом (open source projects). В 2018 году в открытом доступе на хостинге IT-проектов GitHub существовало более 180 000 открытых проектов [22], и их количество продолжает расти.

Открытые лицензии программного обеспечения позволяют разработчикам надлежащим образом использовать, модифицировать и распространять программное обеспечение. За нарушение условий лицензии предусмотрена ответственность, включающая крупные денежные штрафы. Лицензирование открытого программного обеспечения — сложная область со множеством нюансов. На данный момент насчитывается более 450 различных открытых лицензий [27].

Обычно программисты склонны повторно использовать любой находящийся в открытом доступе код, не обращая внимания на его лицензию. В крупных IT-компаниях вопросом лицензирования занимаются специальные юристы, которые проверяют лицензии открытого ПО, использующееся в продуктах компании, и выявляют возможные несовместимости. Вместе с тем, у обычных разработчиков часто не хватает ресурсов для таких пристальных проверок.

Согласно результатам исследования фон Крога и др. [42] существуют две основные причины нарушения лицензий программного обеспечения с открытым исходным кодом. Первая причина заключается в том, что из-за ограничения в ресурсах и времени разработчики часто повторно используют сторонний код, чтобы не тратить силы на решение тривиальных задач. Вторая причина заключается в том, что из-за большого количества открытых лицензий разработчики программного обеспечения часто не понимают их условия и различия между ними, что впоследствии может приводить к нарушениям лицензирования [9].

Для того, чтобы изучить текущую ситуацию с заимствованием кода и нарушением лицензий, лаборатория Методов машинного обучения в программной инженерии JetBrains Research провела исследование потенциальных нарушений лицензирования в популярных Java-проектах

на GitHub [29]. В рамках исследования было изучено и описано значительное количество открытых лицензий, а также сложная система их совместимости. По результатам исследования оказалось, что целых 9,4% методов Java-классов в данных проектах потенциально могли быть скопированы с нарушением лицензирования.

Чтобы помочь разработчикам не совершать подобных ошибок и не тратить время на отдельные проверки, можно показывать по запросу необходимую информацию прямо в интегрированной среде разработки (Integrated Development Environment, IDE). Такая возможность позволит определять совместимые лицензии проекта и сообщать программисту о потенциальных нарушениях. Благодаря этому разработчики смогут избегать ошибок, связанных с лицензиями, что потенциально снизит количество нарушений лицензирования в открытых программных продуктах.

Постановка задачи

Целью данной выпускной квалификационной работы является разработка плагина к IntelliJ IDEA для работы с лицензиями Java-проектов. Плагин должен позволить на основании лицензий компонент проекта определять допустимые лицензии для каждого модуля проекта и сообщать программисту о потенциальных нарушениях.

Для достижения цели были поставлены следующие задачи.

- Сделать обзор предметной области.
- Провести анализ существующих решений в этой области.
- Разработать архитектуру плагина.
- Реализовать алгоритм контроля лицензий проекта:
  - реализовать процедуру извлечения информации о лицензиях компонент проекта;

  - реализовать механизм определения допустимых лицензий для модуля проекта на основании лицензий компонент проекта.

- Реализовать пользовательский интерфейс плагина.
- Провести апробацию созданного инструмента.

1. Обзор предметной области

1.1. Открытое программное обеспечение

Открытое программное обеспечение (open source software) — это программное обеспечение с открытым исходным кодом. Разработка первых открытых проектов началась в конце 70-х–начале 80-х годов. Один из наиболее известных таких проектов — это система компьютерной верстки TeX, разрабатываемая с 1979 года известным ученым и программистом Дональдом Кнутом [14]. Другим проектом стала операционная система GNU [6], разработка которой началась в 1980 году известным программистом Ричардом Столлманом. Однако широкую популярность open source движение начало набирать после создания в 1998 году организации Open Source Initiative [25] (OSI), которая призвана пропагандировать принципы open source разработки и поддерживать открытые проекты.

В рамках работы организации Open Source Initiative было сформулировано определение open source программного обеспечения, которое содержит исчерпывающую информацию о том, какое программное обеспечение можно считать открытым. С полным текстом определения можно ознакомиться на официальном сайте Open Source Initiative.

Программное обеспечение, как и другие формы интеллектуальной собственности, подлежит лицензированию. Далее подробно рассмотрим лицензии программного обеспечения.

1.2. Лицензии программного обеспечения

Лицензия программного обеспечения — это правовой инструмент, определяющий использование, модификацию и распространение программного обеспечения, защищённого авторским правом.

Все лицензии программного обеспечения можно разделить на пять различных типов [32, 40], каждый из которых обладает своим набором разрешений и ограничений использования. Помимо них существует еще статус “общественного достояния”, которое не является лицензией, но

тоже является способом предоставления прав. Различные способы передачи прав представлены на рисунке 1.

Рис. 1: Способы передачи прав на программное обеспечение.

Стоит отметить, что лицензии, входящие в категории “Разрешительные лицензии” и “Копилефтные лицензии” являются открытыми (open source), так как программное обеспечение, лицензированное таким образом, удовлетворяет определению открытого программного обеспечения OSI.

Наиболее исчерпывающий список открытых лицензий можно найти на официальном сайте открытого стандарта для передачи сведений о программном обеспечении The Software Package Data Exchange (SPDX) [27], который включает сведения о лицензиях и авторских правах.

Кратко рассмотрим особенности каждого способа передачи прав.

Общественное достояние

Наиболее разрешительным инструментом для передачи прав является статус общественного достояния (public domain). Делая своё программное обеспечение общественным достоянием, авторы отказываются от всех прав и дают возможность распоряжаться программным продуктом без каких-либо ограничений. В том числе, у третьих лиц появляется возможность сделать это программное обеспечение закрытым или даже продать его. Понятие общественного достояния в разных странах различается, что иногда может приводить к юридическим сложностям. Чтобы этого избежать, используются разрешительные лицензии.

Разрешительные лицензии

Разрешительные (permissive) лицензии предоставляют возможность использовать, модифицировать и распространять соответствующее программное обеспечение, включая возможность лицензировать модификации ПО проприетарными лицензиями. Такие лицензии часто требуют указывать оригинальных авторов программного обеспечения при распространении модифицированного ПО. Помимо этого разрешительные лицензии часто включают в себя отказ от каких-либо гарантий программного обеспечения и снимают любую ответственность с авторов продукта. Наиболее распространенными разрешительными лицензиями [4] являются MIT [21], Apache 2.0 [3] и BSD-3-Clause [1].

Иногда авторы программного обеспечения хотят ограничить возможность лицензирования модификаций продукта. Чтобы это обеспечить, используются копилефтные лицензии.

Копилефтные лицензии

Обычно выделяют два вида копилефтных лицензий: сильные копилефтные (strong copyleft) и слабые копилефтные (weak copyleft) лицензии.

Сильные копилефтные лицензии — это лицензии, которые, как и разрешительные, позволяют свободно модифицировать и распространять модифицированный продукт, однако лицензия модифицированного продукта должна быть такой же, как и на оригинальном ПО. Сильные копилефтные лицензии также требуют указывать оригинальных авторов на модификациях продукта. Таким образом, сильные копилефтные лицензии позволяют авторам ограничить использование своего программного продукта и запретить использование его модификаций под другими лицензиями. Наиболее популярными сильными копилефтными лицензиями являются лицензии семейства GNU General Public License (GPL) [4, 13].

Слабые копилефтные лицензии отличаются от сильных различными послаблениями в ограничениях использования, модификации и распро-

странения программного продукта. Наиболее известным примером таких лицензий являются лицензии семейства GNU Lesser General Public License (LGPL) [13]. Они, так же как и лицензии семейства GPL, разрешают изменять и распространять модифицированный код только при условии сохранения данной лицензии, однако при использовании кода в качестве библиотеки выполнение данного требования не обязательно.

Некоммерческие лицензии

Некоммерческие (noncommercial) лицензии предоставляют права на модификацию и распространение программного обеспечения, но только для некоммерческой деятельности. Часто некоммерческие лицензии являются и копилефтными лицензиями. Примерами лицензий такого типа являются Java Research License (JRL) [19] и Aladdin Free Public License (AFPL) [2].

Проприетарные лицензии

Проприетарные (proprietary) лицензии — это лицензии, которые обеспечивают автору программного обеспечения монополию на его использование, копирование и модификацию. Программное обеспечение, находящееся под данным типом лицензий, является проприетарным. Исходный код таких продуктов обычно является закрытым, но это необязательно.

Коммерческая тайна

Коммерческая тайна (trade secret) — информация, которая имеет коммерческую ценность в силу неизвестности её третьим лицам, к которой нет свободного доступа на законном основании. Программное обеспечение, являющееся коммерческой тайной, нигде не публикуется. Исходный код таких продуктов хранится в тайне, а за разглашение информации о внутреннем устройстве программного обеспечения, попадающего под коммерческую тайну, предусмотрена ответственность.

1.3. Совместимость типов лицензий

Одним из наиболее важных аспектов в области лицензирования программного обеспечения является совместимость различных типов лицензий. В общем случае совместимость типов лицензий представлена на рисунке 2.

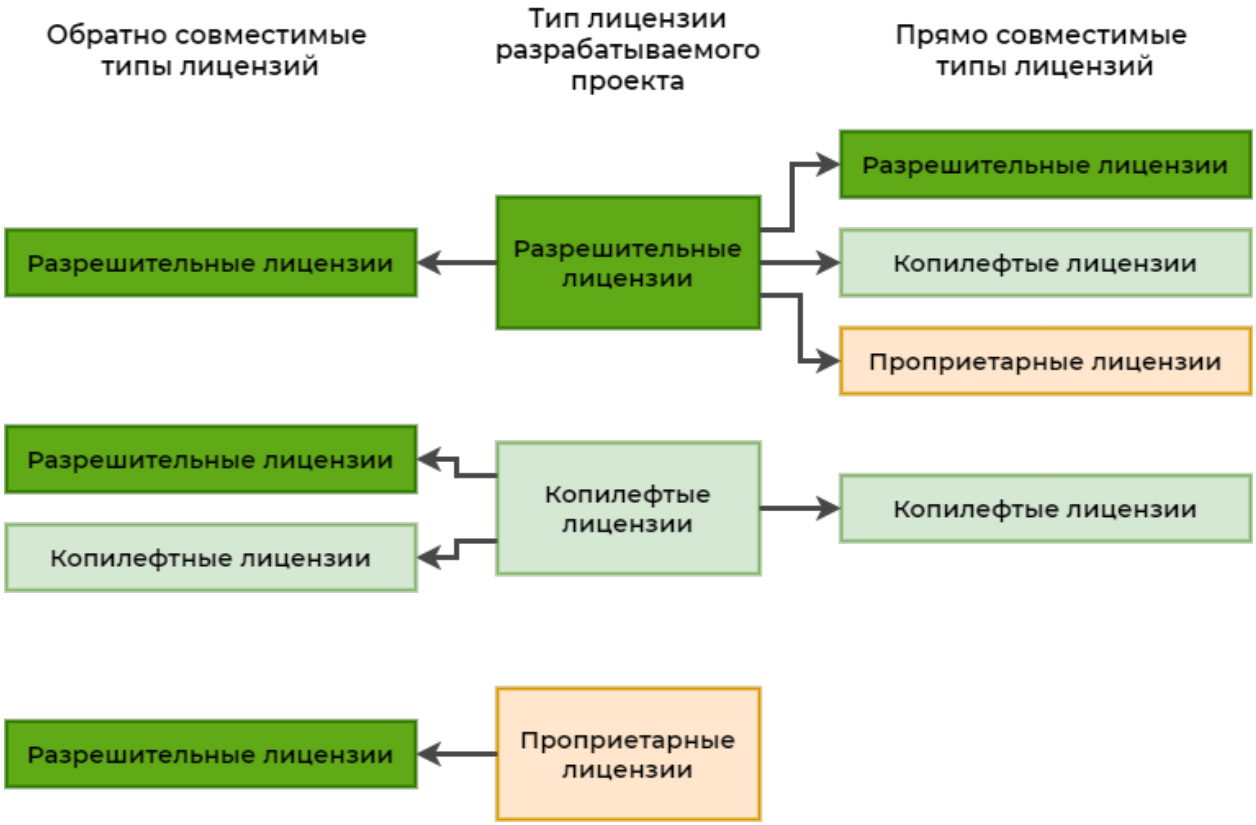

Рис. 2: Совместимость типов лицензий.

Прямая совместимость определяет, какой лицензией могут лицензироваться производные продукты, которые используют программное обеспечение автора. Обратная совместимость определяет, какую лицензию автор может использовать для лицензирования своего программного продукта, исходя из лицензий программного обеспечения, которое он использовал.

Например, в программном проекте, который имеет разрешительную лицензию, можно использовать только программное обеспечение, находящееся под разрешительной лицензией. В то же время модификации такого проекта можно распространять под различными лицензиями, в том числе проприетарными.

Нужно отметить, что на диаграмме отсутствует тип некоммерческих лицензий, так как для него нельзя в общем случае определить совместимость с другими типами лицензий. Совместимость с некоммерческими лицензиями необходимо рассматривать отдельно для каждой лицензии, поэтому в рамках данной работы они рассматриваться не будут.

1.4. Совместимость открытых лицензий

Особое внимание необходимо уделить совместимости открытых лицензий. Важно отметить, что использование сторонней лицензированной библиотеки является повторным использованием кода данной библиотеки, а значит требует соблюдения всех лицензионных ограничений. Поэтому одной из задач разрабатываемого плагина является поиск потенциальных нарушений лицензионных ограничений и уведомление разработчика о найденных нарушениях.

На рисунке 3 детально изображена совместимость наиболее популярных открытых лицензий.

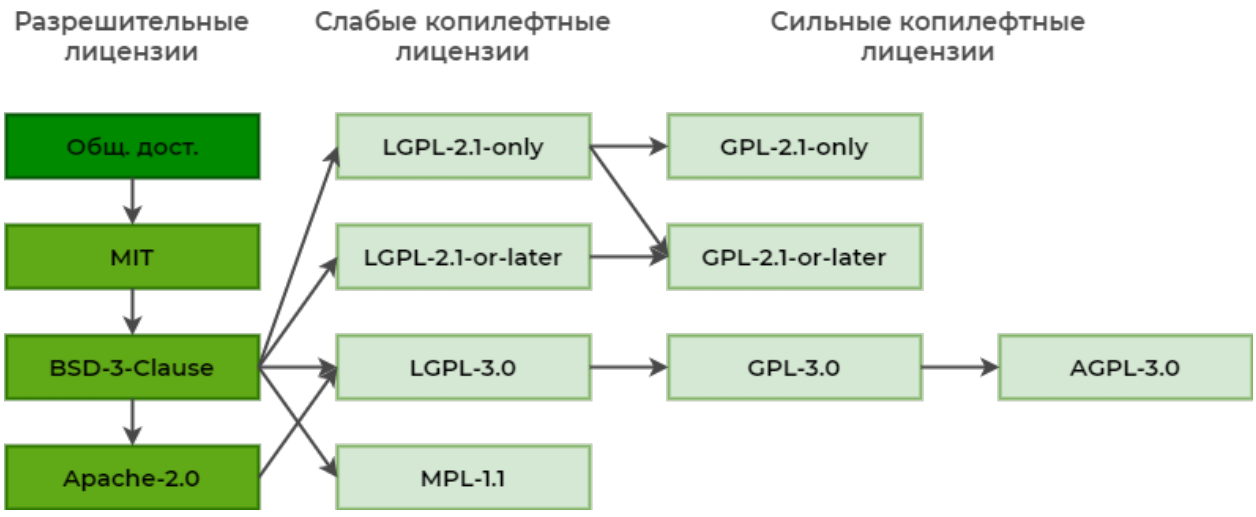

Рис. 3: Совместимость популярных открытых лицензий.

Стрелки между лицензиями обозначают их прямую совместимость. Важно отметить, что каждая рассмотренная на рисунке 3 лицензия совместима сама с собой. Помимо этого на диаграмме подразумевается транзитивная совместимость лицензий. Так например, в продукте с

лицензией GPL-3.0 можно использовать программное обеспечение под лицензией Apache-2.0, но не наоборот.

1.5. Исследования нарушений лицензирования ПО

Существует ряд научных работ, посвященных исследованию нарушений лицензирования в открытом программном обеспечении. Одним из таких исследований является работа Даниеля Германа и др. [15], в результате которой были обнаружены несовместимости лицензий зависимостей пакетов дистрибутива Fedora-12 GNU/Linux. Код под лицензиями GPL часто некорректно использовался в пакетах с более разрешительными лицензиями.

В работе Романского и др. [28] исследованы возможные нарушения лицензий при миграции кода между модулями Python и постами сервиса StackOverflow, а также сделан вывод о том, что посты StackOverflow часто содержат код, лицензия которого несовместима с лицензией платформы.

Другой подобной работой является эмпирическое исследование несовместимости лицензий в открытом программном обеспечении Арунеш Матура и др. [10]. В рамках этого исследования были изучены открытые проекты хостинга репозиториев Google Code, среди которых был обнаружен и подробно описан ряд проблем, связанных с несовместимостью лицензий.

Хунъю Чжан и др. провели исследование, посвященное автоматической проверке совместимости лицензий [33], в ходе которого был разработан инструмент для определения совместимости лицензий проектов на базе сервиса Google Code Search. По результатам экспериментов с данным инструментом было обнаружено несколько новых нарушений лицензий в открытом программном обеспечении.

Перечисленные исследования показывают, что лицензирование открытого программного обеспечения — нетривиальная область. Несмотря на то, что написано множество статей и создан целый ряд инструментов для автоматического определения лицензии файла и обнаружения

несовместимости лицензий, данная область требует дальнейших исследований.

2. Обзор существующих решений

На данный момент существует несколько популярных инструментов, которые предоставляют возможность извлекать информацию о лицензиях из Java-проектов, включая информацию о лицензиях зависимостей проекта и файлов с исходным кодом. Такие инструменты полезны для небольших компаний и независимых разработчиков, которые не имеют достаточно ресурсов для использования услуг профессиональных юристов по работе с лицензиями проектов.

После анализа инструментов для работы с лицензиями проектов были выделены следующие параметры для сравнения:

- компоненты проекта, анализируемые инструментом;
- тип инструмента (консольное приложение, графическое приложение, веб-сервис, плагин для системы сборки проекта);
- способность рекомендации основной лицензии проекта;
- способность определения потенциальных нарушений лицензирования;
- открытость исходного кода.

Возможность извлекать информацию не только об основной лицензии проекта, но и о лицензиях зависимостей и файлов с исходным кодом, непосредственно определяет возможности самого инструмента.

Тип инструмента определяет удобство его использования. Обычному пользователю проще работать с инструментом, который имеет наглядный графический интерфейс или интегрирован с популярной системой сборки проектов. Консольные приложения обычно неудобны для быстрого освоения и использования. Часто их команды имеют множество параметров, что требует времени на их изучение и часто отталкивает программистов от использования таких инструментов.

Несмотря на то, что список лицензий всех компонент проекта сам по себе является полезной информацией для программиста, ему попрежнему необходимо разбираться в выборе корректной лицензии для

своего проекта. Инструменты для работы с лицензиями проектов могут упростить этот процесс и предоставить пользователю подсказки, которые помогут корректно выбрать лицензию и избежать всех сложностей, связанных с лицензированием программного обеспечения.

Определение потенциальных нарушений лицензирования является одной из целей инструментов для работы с лицензиями проектов. Данная возможность позволяет предупредить программиста о нарушениях лицензирования при использовании библиотеки или файлов с исходным кодом, лицензии которых не совместимы с основной лицензией разрабатываемого проекта.

Открытый исходный код инструмента позволяет использовать его бесплатно, что актуально для независимых разработчиков и небольших компаний. Кроме того, у сторонних разработчиков появляется возможность адаптировать код открытого инструмента под свои нужды, что потенциально увеличивает применимость данного инструмента.

В таблице 1 представлено сравнение некоторых популярных инструментов для работы с лицензиями Java-проектов.

Сравнение показало, что большинство инструментов работают только с основными лицензиями проектов и их исходным кодом, не обрабатывая лицензии зависимостей проекта. Ни один из рассмотренных бесплатных инструментов не предоставляет пользователю рекомендации по выбору основной лицензии проекта и не сообщает о несовместимых лицензиях. Кроме того, ни один из инструментов не встроен в интегрированную среду разработки (IDE).

Интеграция инструмента для работы с лицензиями с популярной IDE облегчает установку и настройку инструмента, а также дает возможность программисту пользоваться таким инструментом, не выходя из уже привычного редактора кода, что повышает мотивацию программиста уделять необходимое внимание лицензиям проекта. Благодаря интеграции инструмента в IDE среда разработки может в фоновом режиме сканировать проект на наличие лицензий и отслеживать потенциальные нарушения. Все это позволит программисту удостовериться в том, что все лицензионные требования в проекте соблюдены.

Инструмент	Лицензии	Тип	Рекомендации	Нарушения	Open
askalono [34]	Проект, исходный код	Консольное приложение	-	-	+
FOSSology [12]	Проект, зависимости, исходный код, бинарные файлы	Веб-сервис	+	+	-
go-license-detector [35]	Проект, исходный код	Консольное приложение	-	-	+
license-checker [38]	Проект, исходный код	Консольное приложение	-	-	+
licensee [39]	Проект	Консольное приложение	-	-	+
scancode-toolkit [41]	Проект, исходный код, бинарные файлы	Консольное приложение	-	-	+
FOSSA [11]	Проект, зависимости, исходный код, бинарные файлы	Веб-сервис	+	+	-
License Maven Plugin [37]	Исходный код	Maven плагин	-	-	+
Gradle License Plugin [36]	Проект, зависимости	Gradle плагин	-	-	+
Gradle License Report [16]	Зависимости	Gradle плагин	-	-	+

Таблица 1: Популярные инструменты для работы с лицензиями проектов

3. Архитектура плагина

Плагин License Detector Plugin[1] (LDP) для работы с лицензиями Java проектов в IntelliJ IDEA разработан в рамках проекта исследовательской лаборатории машинного обучения в области программной инженерии JetBrains Research. Важной особенностью разработки данного плагина является нежелательность использования сторонних относительно компании JetBrains инструментов для детектирования лицензий проекта. Поэтому все зависимости проекта являются либо продуктами, либо внутренними разработками компании JetBrains.

LDP написан на языке Kotlin и поддерживает работу с 16 самыми популярными лицензиями Java-проектов. Плагин состоит из двух модулей — Core и Integration. Архитектура плагина представлена на рисунке 4. Оба модуля зависят от компоненты IntelliJ Platform SDK [18] — набора инструментов, необходимых для создания плагинов к платформе IntelliJ Platform [17].

Первый из модулей, Core, содержит внутренние компоненты, реализующие алгоритм контроля лицензий проекта. Компонент Project Manager отвечает за отслеживание состояния проекта и обновление информации о найденных лицензиях компонент проекта. License Manager осуществляет определение совместимых лицензий для каждого модуля проекта, а также выявляет потенциальные нарушения лицензий библиотек и подмодулей каждого модуля проекта. Компонент License Detectors содержит детекторы лицензий, которые определяют тип лицензии на основе её полного текста. Выделение детекторов в отдельную компоненту позволило абстрагироваться от процесса определения лицензий при реализации алгоритма контроля лицензий проекта, а также упростило добавление новых детекторов. Package Search Client отвечает за взаимодействие с Package Search, сервисом для поиска Java библиотек от компании JetBrains. Компоненты Dependency Model и License Model содержат модели данных зависимостей и лицензий соответствен-

[1]Репозиторий плагина License Detector Plugin: https://github.com/JetBrains-Research/license-detector-plugin

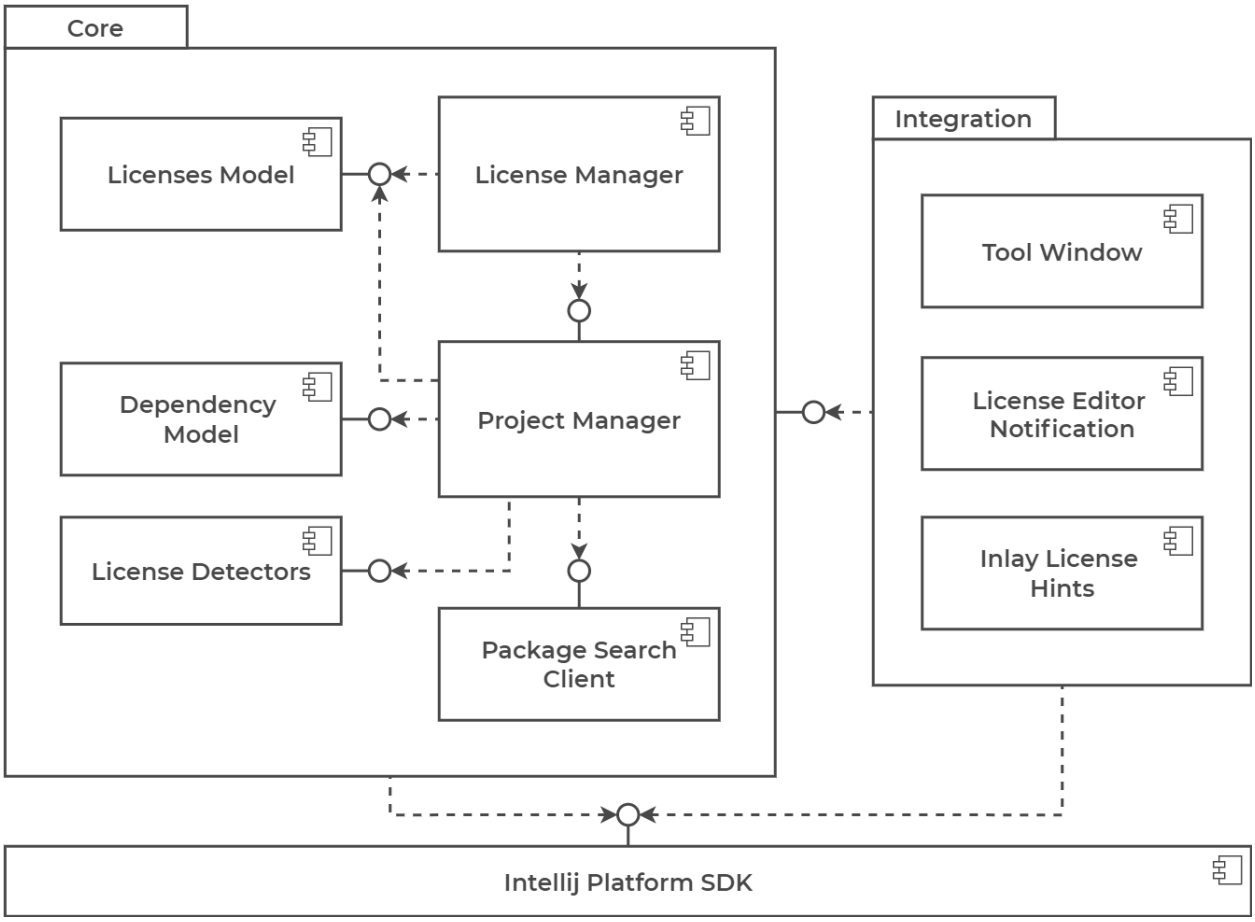

Рис. 4: Архитектура плагина.

но, что позволяет унифицировать работу с различными зависимостями и лицензиями проектов, а также снижают накладные расходы на расширение плагина и поддержку новых лицензий.

Второй модуль, Integration, отвечает за интеграцию с IntelliJ IDEA и предоставляет пользователю графический интерфейс, который позволяет работать с информацией о лицензиях в удобном виде. Компонент Tool Window отвечает за сворачивающееся окно с информацией о лицензиях проекта. License Editor Notification предоставляет панель вверху редактора файла с лицензией, а Inlay License Hints отвечает за подсказки при редактировании скриптов систем для сборки проектов.

Таким образом, разработанная архитектура позволяет отделить логику работы с лицензиями проекта от пользовательского интерфейса, а также упрощает добавление новых детекторов лицензий и поддержку новых типов лицензий в плагине.

4. Алгоритм контроля лицензий

Алгоритм контроля лицензий отслеживает лицензии компонент проекта и определяет допустимые лицензии для каждого модуля. Допустимые лицензии модуля — это лицензии, которые мог бы иметь модуль и которые не нарушают лицензии библиотек и лицензии других модулей проекта. На основании допустимых лицензий модулей алгоритм также находит потенциальные нарушения лицензирования в проекте. Этапы алгоритма представлены на рисунке 5.

Рис. 5: Алгоритм контроля лицензий.

Первым этапом является извлечение информации о лицензиях компонент проекта. На этом этапе происходит сканирование модулей и библиотек проекта, а также поиск и извлечение информации об их лицензиях.

На втором этапе происходит обработка полученной информации о лицензиях модулей и библиотек, определяются допустимые лицензии для каждого модуля проекта и выявляются потенциальные нарушения лицензий библиотек и подмодулей.

Рассмотрим подробнее процесс извлечения информации о лицензиях компонент проекта.

4.1. Извлечение информации о лицензиях компонент проекта

Извлечение информации о лицензиях происходит из модулей и библиотек проекта. Полученная информация используется для определения допустимых лицензий для каждого модуля проекта и поиске потенциальных нарушений лицензирования. Процесс извлечения информации о лицензиях компонент проекта представлен на рисунке 6.

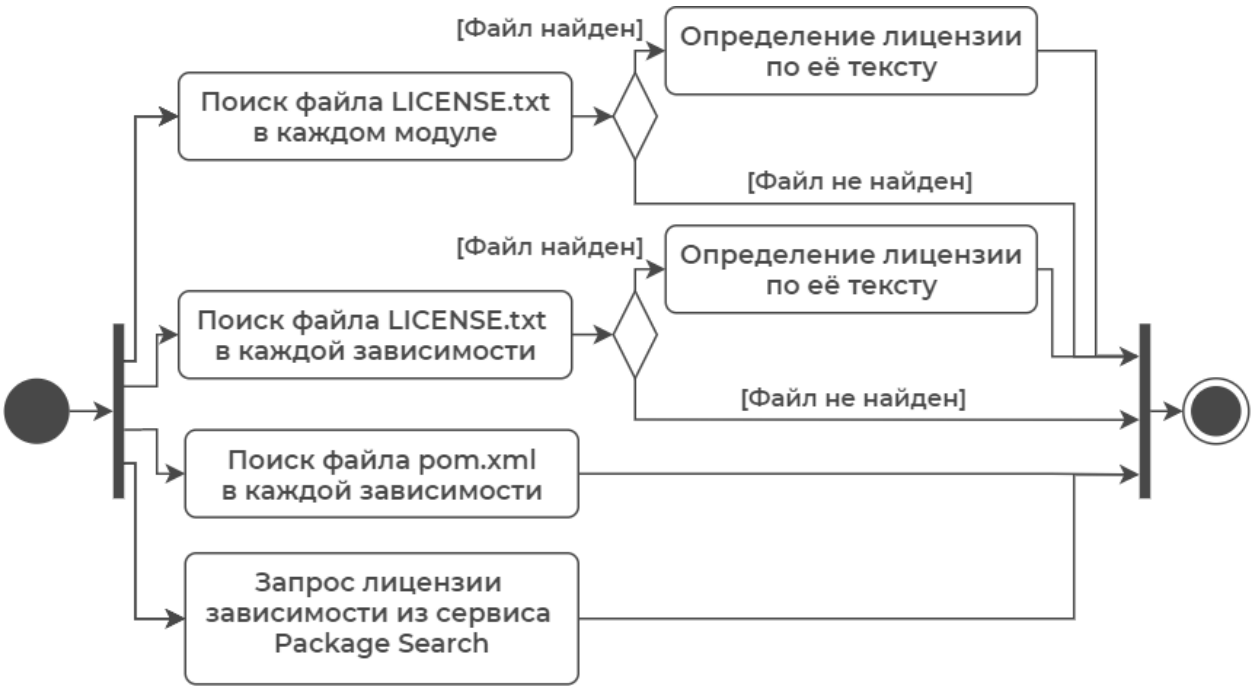

Рис. 6: Процесс извлечения информации о лицензиях компонент проекта.

Извлечение лицензий модулей

Извлечение информации о лицензиях модулей проекта происходит следующим образом. Из IntelliJ Platform извлекается информация о всех модулях проекта. Далее просматриваются корневые директории каждого модуля на наличие файла с названием *LICENSE.txt* или подобным ему. Найденные файлы должны содержать полный текст лицензии, который используется для определения типа лицензии. Процесс определения типа лицензии по её полному тексту описан далее. В

результате мы получаем список модулей проекта с найденными лицензиями.

Необходимо отметить, что использование IntelliJ Platform для получения данных о модулях проекта позволяет отказаться от определения лицензий для каждой директории проекта и определять лицензии только для модулей. Для этого необходимо просматривать только корневые директории каждого модуля в поисках лицензии. Таким способом удалось сделать процесс определения лицензий более оптимальным, чем просмотр всех директорий проекта.

Помимо этого IntelliJ Platform используется для получения информации о библиотеках модуля, что позволяет учесть абсолютно все библиотеки модуля при определении допустимых лицензий и поиске потенциальных нарушений лицензирования. Без использования платформы получение информации о всех библиотеках является нетривиальной задачей. Например, если проект использует систему сборки Gradle, то анализ Gradle скрипта не позволит выявить все библиотеки, подключаемые с помощью Gradle плагинов или пользовательских задач Gradle. IntelliJ Platform лишена подобных недостатков. Поэтому её использование дает плагину преимущество по сравнению с другими инструментами, повышает точность определения потенциальных нарушений лицензирования и корректность рекомендаций лицензий.

Извлечение лицензий библиотек

После извлечения лицензий из модулей происходит извлечение лицензий библиотек проекта, которые хранятся в виде jar-архивов. В таком архиве содержатся все необходимые файлы для работы библиотеки. Среди этих файлов может находится файл *LICENSE.txt* с полным текстом лицензии библиотеки, по которому можно определить тип лицензии. Помимо этого в jar-архиве должен находиться файл *pom.xml*, который содержит в себе метаинформацию о библиотеке, включая название её лицензии. Если такой файл в архиве есть, то извлечение информации о лицензии зависимости сводится к определению лицензии по её названию. Это не всегда просто сделать. На данный момент нет стро-

го формализованных названий лицензий, поэтому название одной и той же лицензии может различаться в метаданных различных библиотек. Для распознавания лицензии по её названию в плагине используются регулярные выражения.

Однако на практике довольно часто оказывается так, что библиотеки поставляются без файла с метаинформацией или в таком файле отсутствует информация о лицензии зависимости. В таком случае узнать о лицензии библиотеки без запросов к сторонним ресурсам практически невозможно.

На данный момент существует несколько репозиториев зависимостей и сервисов для хостингов IT-проектов, которые могут содержать информацию о лицензии конкретной библиотеки. Реализация извлечения информации о лицензиях библиотек из всех этих источников сложна и требует много ресурсов на разработку и отладку.

Альтернативным решением является использование сервиса для поиска Java библиотек Package Search [26] для получения информации о лицензиях библиотек. Сервис индексирует популярные репозитории и хостинги Java библиотек и предоставляет API для получения полной информации о конкретной зависимости, включая лицензию библиотеки и, например, ссылку на репозиторий на GitHub. Package Search является продуктом компании JetBrains, поэтому не является сторонним решением по отношению к разрабатываемому плагину.

Для определения лицензии библиотеки с помощью сервиса Package Search был реализован компонент Package Search Client, который отправляет запросы сервису и преобразовывает полученную информацию во внутреннее представление плагина.

Несмотря на все удобства работы с Package Search, у него есть недостатки. Первый недостаток состоит в том, что сервис не индексирует репозиторий bintray/jcenter [5], в котором опубликовано множество Java-библиотек, которых нет в других репозиториях. Этот недостаток перестал быть актуальным в связи с новостью о скором закрытии репозитория bintray/jcenter [30]. Все библиотеки, которые в нем находились, должны в скором времени мигрировать в другие репозитории, напри-

мер в Maven Central [23] или на GitHub.

Другой недостаток заключается в том, что Package Search предоставляет информацию о лицензии библиотеки в целом, а не для каждой конкретной версии. Если разные версии библиотеки находятся под разными лицензиями, то сервис возвращает список лицензий без уточнения, какая лицензия какой версии соответствует.

Также необходимо отметить, что алгоритм контроля лицензий при поиске потенциальных нарушений учитывает все предоставленные сервисом лицензии. Это позволяет избежать некоторых ситуаций неправильного определения лицензии библиотеки, которые могут приводить к ненайденным реальным нарушениям лицензий. Вместе с тем, использование всех лицензий увеличивает количество ложных потенциальных нарушений, которые находит плагин. Однако в данном случае не пропустить реальное нарушение важнее, чем потенциально снизить количество ложноположительных срабатываний.

Определение лицензии по полному тексту

Отдельно необходимо рассмотреть процесс определения лицензии по её полному тексту. Данный процесс представлен на рисунке 7.

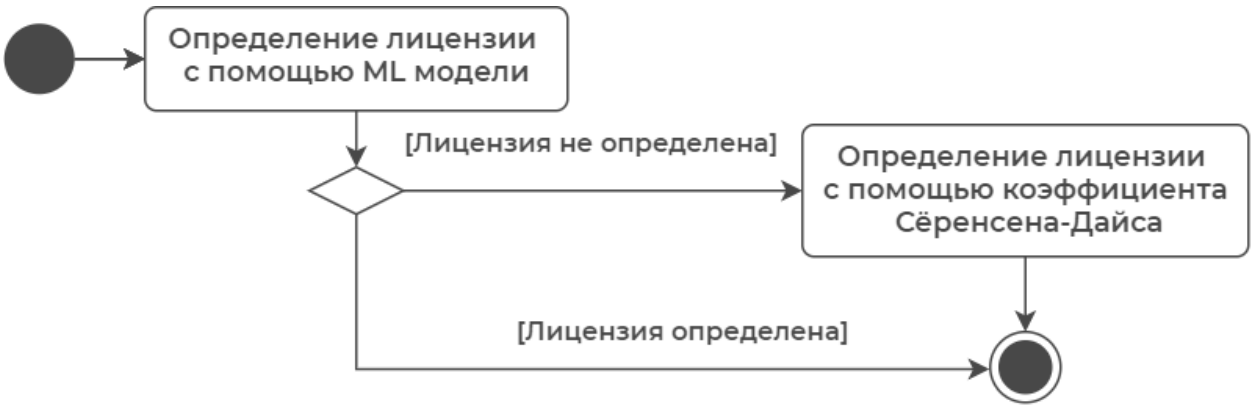

Рис. 7: Процесс определения лицензии по её полному тексту.

Для определения лицензии по её тексту используется два детектора. Первый детектор (ML) использует модель машинного обучения, разработанную студентом Санкт-Петербургского Политехнического университета Иваном Кузнецовым. Модель написана на языке Python с по-

мощью библиотеки CatBoost [7]. Для использования модели в Kotlin плагине, готовая модель была экспортирована в формате ONNX [24] и интегрирована в плагин с помощью библиотеки KInference [20] от компании JetBrains. Модель поддерживает работу с 12 лицензиями. Данное ограничение вытекает из обучающего набора данных, который был размечен только для 12 лицензий.

Для определения остальных лицензий и случаев, когда ML детектор не смог определить тип лицензии, используется детектор на основе коэффициента подобия Сёренсена-Дайса (DSC) [8]. Коэффициент подобия для двух наборов элементов X и Y определяется как:

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

где $|X|$ и $|Y|$ — соответствующие мощности наборов. Чем ближе значение коэффициента к единице, тем больше наборы элементов похожи.

Детектор на основе коэффициента подобия работает следующим образом. Каждая поддерживаемая лицензия содержит заранее подготовленный набор слов оригинального текста. Детектор принимает на вход текст неизвестной лицензии и разбивает его на слова. Полученный набор слов сравнивается с помощью коэффициента Сёренсена-Дайса с наборами слов оригинальных лицензий. Совпадение засчитывается, когда коэффициент подобия больше порогового значения, которое равняется 0.95. Таким образом, детектор требует совпадения с оригинальным текстом лицензии на 95% и допускает небольшие изменения текста.

Данный коэффициент подобия и значение порога подобия выбраны не случайно. Коэффициент Сёренсена-Дайса с порогом 0.95 лежит в основе инструмента для определения лицензий licensee [39], который используется для определения лицензий проектов на популярном хостинге GitHub. Поэтому данный метод определения лицензий является проверенным решением и показывает хороший результат.

Оба детектора лицензий содержатся в компоненте License Detectors модуля Core. Благодаря им разработанный плагин поддерживает работу с 16 лицензиями. В таблице 2 представлены поддерживаемые ли-

цензии и методы их определения в плагине.

Лицензия	Метод определения
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
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

Таблица 2: Поддерживаемые лицензии и методы их определения в плагине

Данные лицензии покрывают примерно 95% всех Java-проектов. Лицензии были выбраны на основе результатов исследования [29], в котором содержится информация о самых популярных лицензиях Java-проектов на GitHub. Несмотря на то, что в исследовании рассматриваются популярные лицензии семейства GPL и LGPL с приставкой or-later, в плагине поддержаны более строгие версии этих лицензий с приставкой only. Лицензии с приставкой or-later, в отличии от лицензий с приставкой only, позволяют использовать лицензированное ПО под той же лицензией более старшей версии. Например, библиотеку с лицензией GPL-2.0-or-later можно использовать при разработке продукта с лицензией GPL-3.0-only, а лицензия GPL-2.0-only не предоставляет такой возможности. Тексты версий only и or-later очень похожи и могут различаться всего в одном словосочетании, что плохо сказывается на точности детектирования. Небольшие различия в текстах таких

лицензий влекут различные правила совместимости с более старшими версиями лицензий. Поэтому для избежания некорректной работы с лицензиями семейств GPL и LGPL плагин поддерживает работу с более строгими версиями этих лицензий.

4.2. Определение допустимых лицензий для модуля проекта

На основе информации о лицензиях компонент, полученной на предыдущем этапе, происходит процесс определения допустимых лицензий для каждого модуля и поиск потенциальных нарушений лицензирования.

На данном этапе для каждого модуля определена лицензия самого модуля, а также лицензии его подмодулей и библиотек. Если у модуля нет лицензии, то используется лицензия модуля родителя, если он существует. Также, если у подмодуля нет лицензии, то все его подмодули и библиотеки учитываются при обработке модуля родителя.

Процесс определения допустимых лицензий представлен на рисунке 8.

Набор допустимых лицензий модулей определяется следующим образом. Для каждой поддержанной в плагине лицензии заданы два набора совместимых с ней лицензий. Один из них предназначен для лицензий библиотек модуля. Он содержит лицензии модуля, которые не будут нарушать данную лицензию, если она является лицензией библиотеки. Такой набор извлекается из каждой лицензии библиотек модуля. В результате получаем множество наборов допустимых лицензий. Пересекая все наборы совместимых лицензий получаем один набор из лицензий, которые не нарушают ни одной лицензии библиотеки этого модуля.

Подобная процедура происходит и для лицензий подмодулей. Для этого используется другой набор совместимых лицензий. Он содержит допустимые лицензии модуля, которые не будут нарушать данную лицензию, если она является лицензией подмодуля. Такой набор извлека-

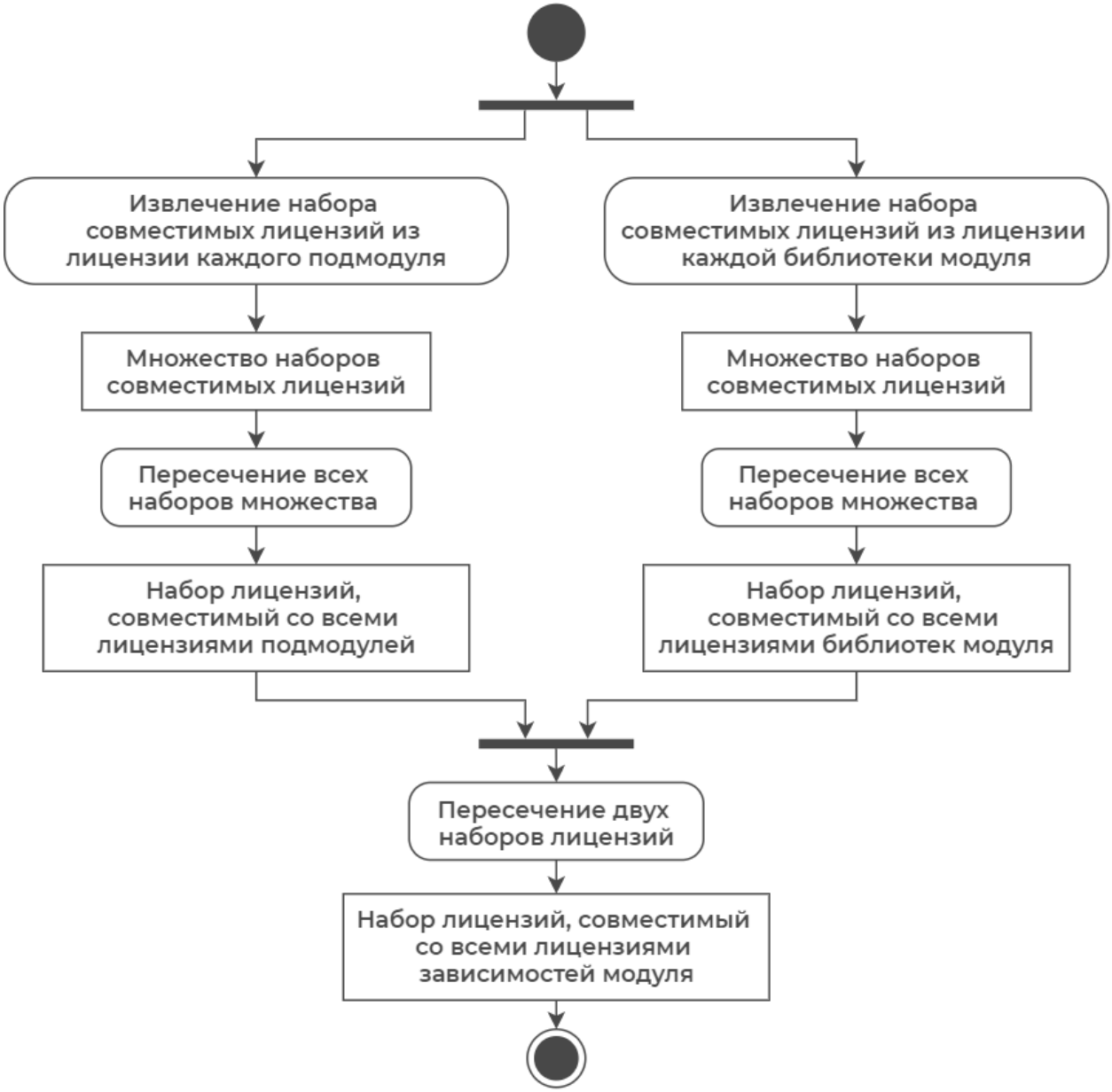

Рис. 8: Процесс определения допустимых лицензий модуля.

ется из каждой лицензии подмодулей. В результате получаем множество наборов совместимых лицензий. Пересекая все наборы совместимых лицензий получаем один набор из лицензий, которые не нарушают ни одной лицензии подмодуля этого модуля.

Полученные два набора лицензий снова пересекаются. В результате получаем набор из допустимых лицензий модуля, которые не нарушают ни лицензии библиотек, ни лицензии подмодулей. Найденные допустимые лицензии модуля позволяют предоставить рекомендацию по выбору лицензии и помочь пользователю выбрать лицензию, не заду-

мываясь о нарушениях лицензирования.

Поиск потенциальных нарушений

Отдельно необходимо рассмотреть поиск потенциальных нарушений лицензирования. Процесс поиска представлен на рисунке 9.

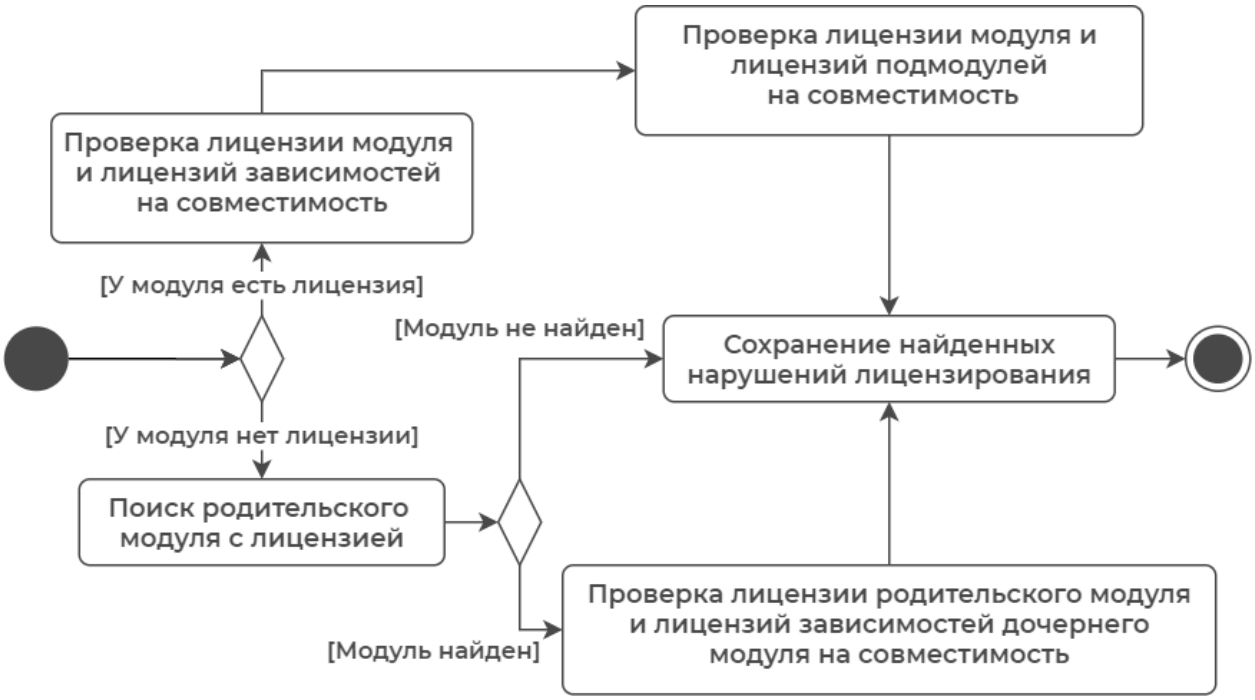

Рис. 9: Процесс поиска потенциальных нарушений лицензирования.

Потенциальные нарушения лицензий определяются как между лицензией модуля и лицензией библиотеки, так и между лицензией модуля и лицензией подмодуля. Процесс поиска потенциальных нарушений во многом схож с определением допустимых лицензий модуля. Для определения нарушения между лицензией модуля и лицензией библиотеки из лицензии библиотеки извлекается набор допустимых лицензий модуля. Этот набор содержит все лицензии, которые может иметь модуль и которые не будут нарушать лицензию библиотеки. Если текущая лицензия модуля входит в этот набор, то нарушения лицензии нет. Если текущая лицензия в набор не входит, то плагин регистрирует потенциальное нарушение лицензии и выводит разработчику соответствующую информацию. Подобная процедура проводится для каждой библиотеки

каждого модуля. Таким образом плагин определяет все потенциальные нарушения связанные с нарушением лицензий библиотек.

Рассмотрим пример определения нарушения лицензии библиотеки. Пример представлен на рисунке 10.

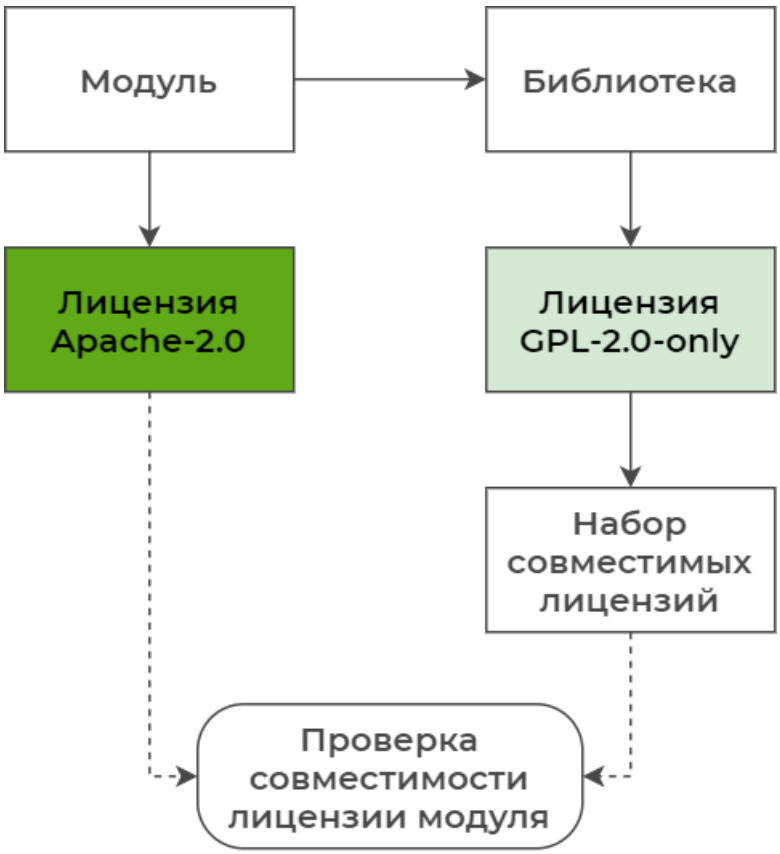

Рис. 10: Пример определения нарушения лицензии библиотеки.

Пусть модуль с лицензией Apache-2.0 использует библиотеку с лицензией GPL-2.0-only. Из лицензии GPL-2.0-only извлекается набор возможных лицензий модуля, которые не нарушают лицензию библиотеки. Плагин обнаруживает, что лицензия Apache-2.0 не входит в этот набор лицензий и регистрирует соответствующее нарушение лицензии.

Аналогичным образом происходит определение потенциального нарушения между лицензий модуля и лицензией подмодуля. Для этого из лицензии подмодуля извлекается набор допустимых лицензий. Этот набор содержит все лицензии, которые может иметь модуль и которые не будут нарушать лицензию подмодуля. Если текущая лицензия модуля входит в этот набор, то нарушения нет. Если текущая лицензия в

набор не входит, то плагин регистрирует потенциальное нарушение лицензии и выводит разработчику соответствующую информацию. Данная процедура проводится для каждого подмодуля каждого модуля. В результате плагин определяет все потенциальные нарушения связанные с нарушением лицензий подмодулей.

Рассмотрим пример определения нарушения лицензии подмодуля. Пример представлен на рисунке 11.

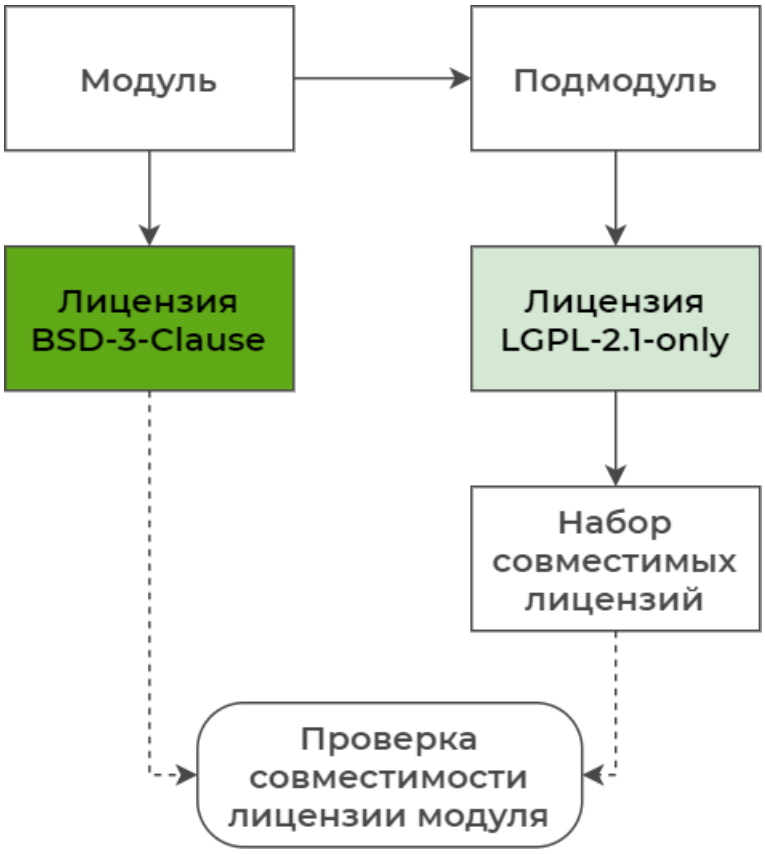

Рис. 11: Пример определения нарушения лицензии подмодуля.

Пусть модуль с лицензией BSD-3-Clause имеет подмодуль с лицензией LGPL-2.1-only. Из лицензии LGPL-2.1-only извлекается набор возможных лицензий модуля, которые не нарушают лицензию подмодуля. Плагин обнаруживает, что лицензия BSD-3-Clause не входит в этот набор лицензий и регистрирует нарушение лицензии подмодуля.

5. Пользовательской интерфейс плагина

Разработанный плагин предоставляет пользователю графический интерфейс для удобства работы с лицензиями проекта. Для реализации интерфейса использована библиотека Swing [31], которая является стандартной библиотекой для разработки графического интерфейса в плагинах к IntelliJ IDEA и входит в IntelliJ Platform SDK.

Рассмотрим подробнее реализованные компоненты графического интерфейса.

Tool Window

Основным графическим интерфейсом плагина является сворачивающееся окно Tool Window. Окно содержит две вкладки и предоставляет основную информацию о лицензиях проекта.

Первая вкладка, Project License, содержит информацию о лицензии корневого модуля проекта с подробным описанием разрешений и ограничений лицензии. Кроме того, вкладка содержит потенциальный список нарушений лицензирования в проекте, которые плагин смог обнаружить. Project License также предоставляет возможность экспортировать список нарушений лицензирования в формате *json* и перезапустить алгоритм контроля лицензий проекта. Данная вкладка представлена на рисунке 12.

Вторая вкладка, Package License, проставляет информацию о найденных библиотеках проекта и их лицензиях. Также на этой вкладке есть возможность фильтрации зависимостей по принадлежности к модулю и по поисковому запросу. Вкладка Package License представлена на рисунке 13.

License Editor Notification

Помимо окна Tool Window, плагин предоставляет панель License Editor Notification. Данная панель находится вверху редактора файлов с лицензией. Она позволяет посмотреть, какие из лицензий совмести-

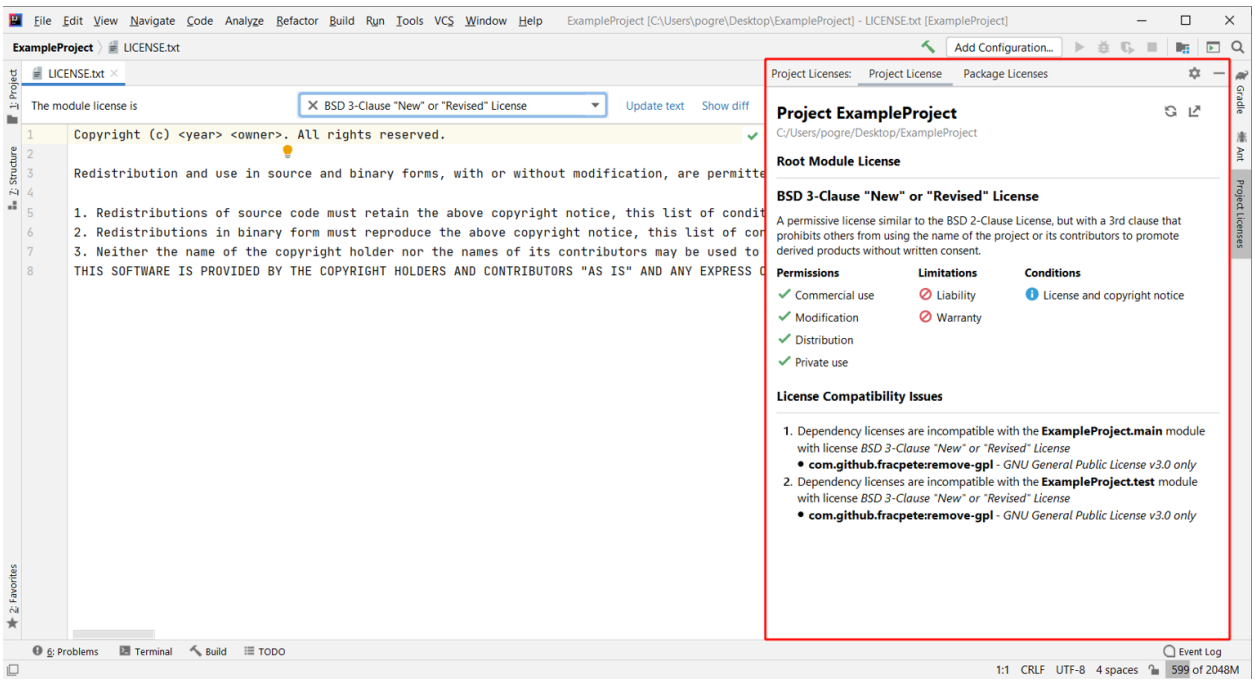

Рис. 12: Вкладка Project License.

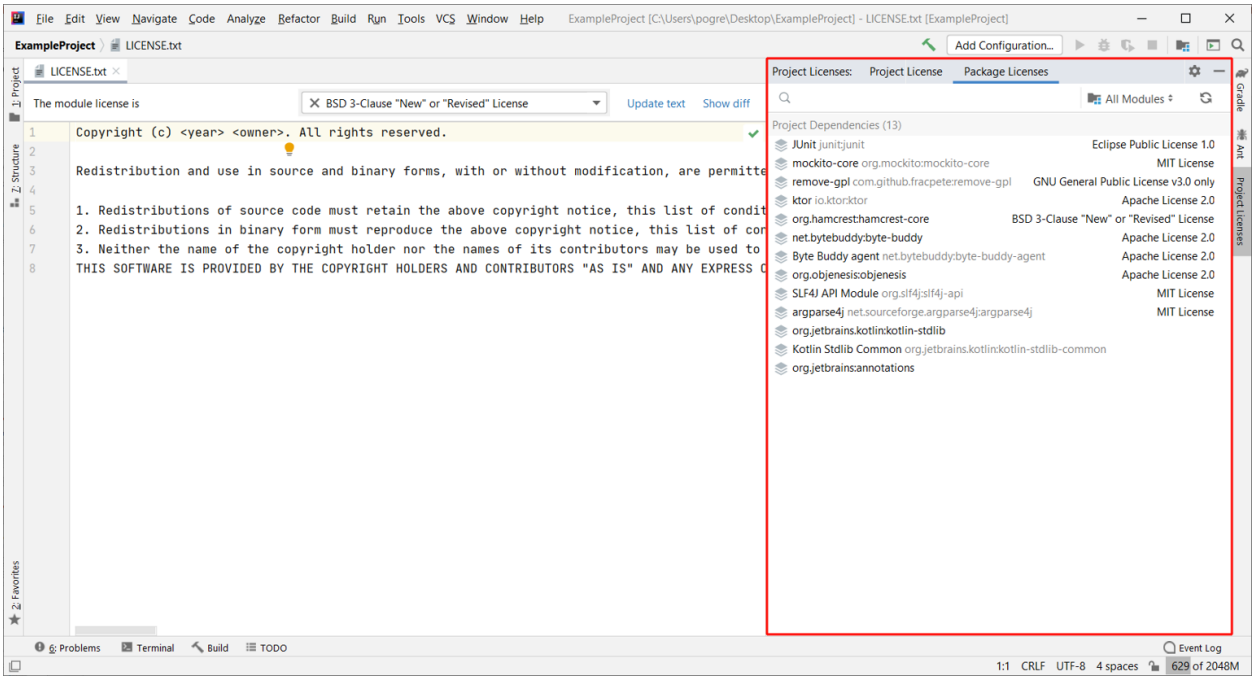

Рис. 13: Вкладка Package License.

мы со всеми лицензиями зависимостей модуля или проекта и выбрать наиболее подходящую. Кроме того, панель предоставляет возможность сравнить текущий текст файла лицензии с оригинальным текстом лицензии. Панель License Editor Notification представлена на рисунке 14.

В данном примере у проекта есть зависимость с лицензией GPL-

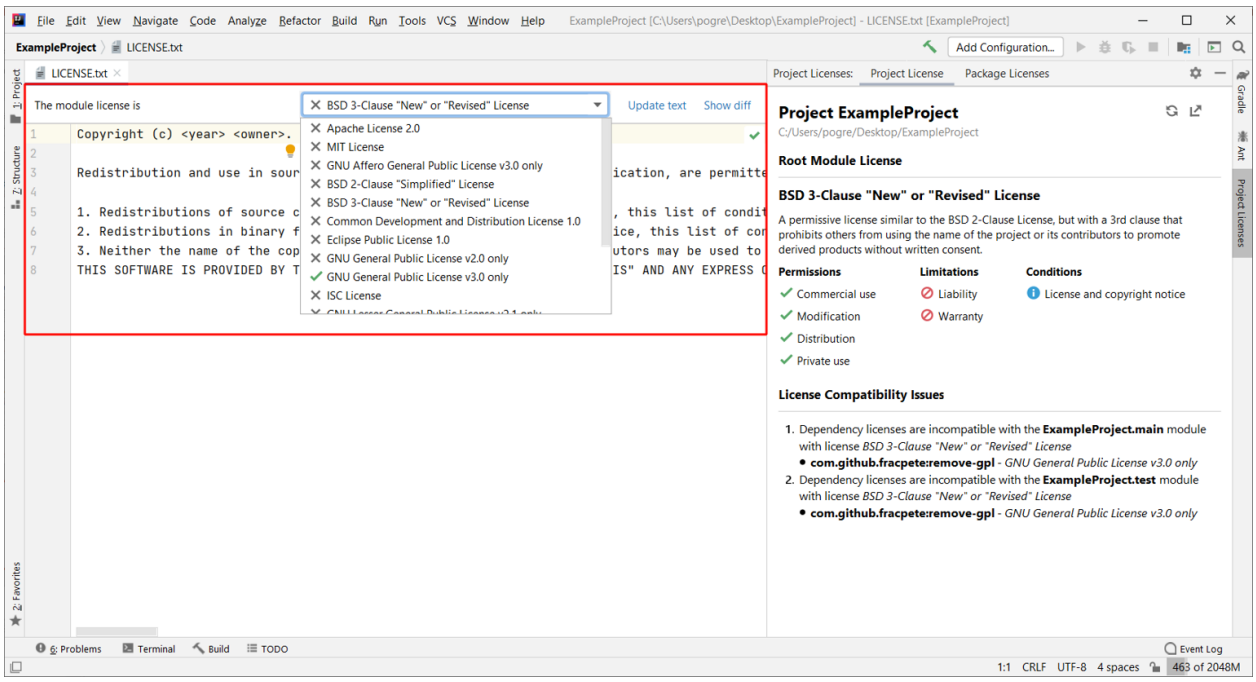

Рис. 14: Панель в редакторе для работы с файлом основной лицензии проекта.

3.0, поэтому совместимой с ней лицензией для всего проекта является только лицензия GPL-3.0, что и отмечено галочкой. Для того, чтобы узнать, какие нарушения возникают при использовании несовместимых лицензий, необходимо выбрать такую лицензию из списка лицензий. После этого в окне ToolWIndow отобразится соответствующий список потенциальных нарушений.

Помимо этого плагин предоставляет возможность создать файл с наиболее разрешающей допустимой лицензией модуля. Благодаря этим возможностям пользователь, который практически не имеет знаний о лицензиях, сможет выбрать корректную лицензию для своего проекта и не допустить ошибку.

Inlay License Hints

Разработанный плагин также предоставляет подсказки с лицензиями библиотек при редактировании скрипта системы сборки проекта. Напротив каждой команды с подключением библиотеки появляется подсказка с названием лицензии этой библиотеки. Такие подсказ-

ки позволяют пользователю сразу узнать лицензию зависимости прямо в редакторе кода. Данные подсказки реализованы для Maven, Groovy Gradle и Kotlin Gradle скриптов. Пример работы подсказок в Kotlin Gradle скрипте представлен на рисунке 15.

Рис. 15: Подсказки лицензий библиотек в Kotlin Gradle скрипте.

6. Апробация плагина

В рамках апробации плагина были изучены потенциальные нарушения лицензирования, которые находит плагин, а также проанализированы самые частые найденные нарушения между лицензиями.

Апробация плагина проводилась на 100 Java-проектах на GitHub с наибольшим количеством звёзд. Для запуска плагина на множестве проектов был написан специальный скрипт, который запускает IntelliJ IDEA в headless режиме (без графического интерфейса) и собирает информацию о нарушениях лицензий из плагина. Однако в процессе апробации оказалось, что часть проектов некорректно импортируются с помощью фреймворка для тестирования IntelliJ Platform. Поэтому такие проекты были обработаны вручную и открыты в IDE в стандартном режиме.

По результатам апробации в 100 проектах плагин обнаружил 762 потенциальных нарушения между лицензией библиотеки и лицензией модуля. Десять самых популярных пар лицензий в потенциальных нарушениях лицензирования представлены на рисунке 16.

Среди найденных нарушений лицензий выделяется группа из 314 случая использования библиотек с разрешительными лицензиями (Apache-2.0, MIT, BSD-3-CLause) в модулях без лицензии (No license). Данные потенциальные нарушения связаны с нестандартными или модифицированными текстами лицензий модулей, что не позволяет программно определить тип лицензии. Тем не менее, данные нарушения указывают разработчику на возможные проблемы с лицензией модуля и могут мотивировать его поменять лицензию на одну из общепринятых.

Помимо этого среди обнаруженных нарушений 384 потенциальных нарушения связаны с использованием библиотек под лицензией GPL-2.0-only в модулях с разрешительными лицензиями (Apache-2.0, MIT и BSD-3-CLause). Причиной этих потенциальных нарушений могла стать строгая работа с модификациями лицензии GPL-2.0. Библиотеки с лицензией GPL-2.0-with-classpath-exception могут использоваться в про-

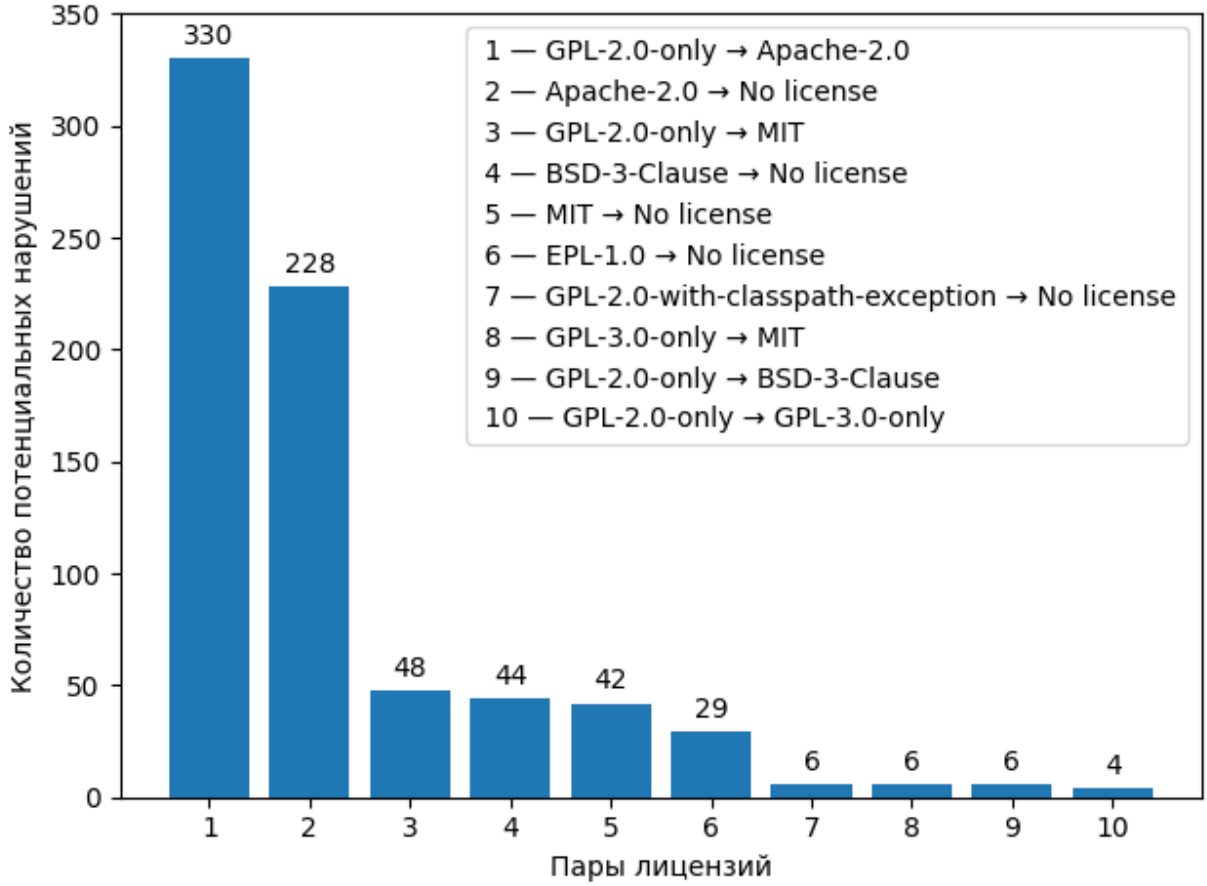

Рис. 16: Десять самых популярных пар лицензий (библиотека → модуль) в потенциальных случаях нарушения правил лицензирования.

екте с разрешительной лицензией, например с Apache-2.0, в отличии от библиотек с лицензией GPL-2.0-only. Однако тексты лицензий GPL-2.0-with-classpath-exception и GPL-2.0-only могут различаться всего в одном предложении. В случаях, когда нет возможности определить точную модификацию лицензии, плагин выбирает наиболее ограничивающую GPL-2.0-only, что приводит к появлению потенциальных нарушений. Такой процесс работы с модификациями лицензий GPL-2.0 позволяет не пропустить реальные нарушения лицензии GPL-2.0-only, обращает внимание разработчика на подозрительные библиотеки и побуждает проверить корректность найденных нарушений вручную.

Необходимо отметить, что некоторые найденные потенциальные нарушения после тщательной ручной проверки могут оказаться ложными и не будут являться нарушениями как таковыми. Однако одной из

задач плагина является задача отсеивания заведомо корректных сочетаний лицензий и предоставления разработчику информации о подозрительных случаях сочетания лицензий. Разработчик, получив информацию о потенциальных нарушениях, может вручную проверить их на корректность или устранить нарушение сразу, отказавшись от библиотеки с несовместимой лицензией или изменив лицензию соответствующего модуля.

Также необходимо отметить, что плагин не обнаружил потенциальных нарушений лицензий между модулями самого проекта. Вероятно это связано с тем, что разработчики обычно лицензируют весь проект целиком, а не каждый модуль по отдельности. Поэтому нарушений лицензирования между модулями проекта не возникает.

Заключение

В ходе данной работы были получены следующие результаты.

1. Сделан обзор предметной области. Рассмотрены различные типы открытых лицензий и правила совместимости между ними. Также сделан обзор известных исследований о нарушении лицензирования при копировании кода открытых проектов.

2. Проведен анализ популярных инструментов для работы с лицензиями проектов, который показал, что большинство инструментов не обрабатывают лицензии зависимостей проекта. Бесплатные инструменты не предоставляют пользователю рекомендации по выбору основной лицензии проекта и не сообщают о несовместимых лицензиях. Кроме того, ни один из инструментов не встроен ни в одну из популярных интегрированных сред разработки.

3. Спроектирована архитектура плагина. Она позволила отделить логику работы с лицензиями от пользовательского интерфейса и упростить поддержку новых детекторов и лицензий в плагине.

4. Реализован алгоритм контроля лицензий проекта. Реализован механизм извлечения информации о лицензиях модулей и зависимостей проекта, а также реализован механизм определения совместимых лицензий для каждого модуля проекта на основании лицензий зависимостей и подмодулей.

5. Реализован интерфейс плагина, который содержит подробную информацию о лицензиях модулей и зависимостей, а также список потенциальных нарушений лицензирования. Кроме того, интерфейс содержит рекомендации по выбору лицензии, совместимой с лицензиями всех модулей и зависимостей проекта.

6. Проведена апробация разработанного плагина на 100 самых популярных проектах на GitHub, содержащих код на Java. Плагин

обнаружил 762 потенциальных нарушений лицензирования, из которых 384 потенциальное нарушение связано с использованием библиотек с модификациями лицензии GPL-2.0 в модулях с разрешительной лицензией.

В результате данной работы была реализована и выпущена пилотная версия плагина для работы с лицензиями Java-проектов, который может успешно применяться для поиска потенциальных нарушений лицензирования. Плагин может быть улучшен главным образом за счет расширения списка поддерживаемых лицензий и более точных инструментов для определения типа лицензии по её полному тексту. Весь код проекта находится в открытом доступе на GitHub[2].

[2]Репозиторий плагина License Detector Plugin: https://github.com/JetBrains-Research/license-detector-plugin

Благодарности

Я хочу поблагодарить Ярослава Голубева за всестороннюю поддержку в процессе работы над плагином, а также за помощь с корректурой текста и ценные замечания по простоте восприятия работы.

Также я хочу поблагодарить Владислава Танкова за помощь в решении проблем, связанных с технической частью этой работы.

Отдельное спасибо Ивану Кузнецову, который разработал модель машинного обучения для определения лицензии по тексту и помогал с интеграцией модели в плагин.

Кроме того, я выражаю свою благодарность Егору Булычеву за то, что согласился отрецензировать эту работу и дал ценные замечания.

И, конечно же, я хочу поблагодарить своего научного руководителя, Тимофея Брыксина, который качественно и трудолюбиво оказывает помощь своим студентам в написании выпускных квалификационных работ.

Список литературы

[1] The 3-Clause BSD License. — 1999. — Режим доступа: https://opensource.org/licenses/BSD-3-Clause (дата обращения: 29.11.2020).

[2] Aladdin Free Public License. — 2000. — Режим доступа: https://www.tug.org/fonts/getnonfreefonts/AFPL.html (дата обращения: 29.11.2020).

[3] Apache License, Version 2.0. — 2004. — Режим доступа: https://opensource.org/licenses/Apache-2.0 (дата обращения: 29.11.2020).

[4] Balter Ben. Open source license usage on GitHub.com. — 2015. — Режим доступа: https://github.blog/2015-03-09-open-source-license-usage-on-github-com/ (дата обращения: 29.11.2020).

[5] Bintray JCenter dependency repository. — 2021. — Режим доступа: https://bintray.com/bintray/jcenter (дата обращения: 29.03.2020).

[6] 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. — ISBN: 9781680503012. — Access mode: https://books.google.ru/books?id=ley-tAEACAAJ.

[7] CatBoost — gradient boosting on decision trees. — 2021. — Режим доступа: https://catboost.ai/ (дата обращения: 29.03.2020).

[8] Dice Lee R. Measures of the Amount of Ecologic Association Between Species // Ecology. — 1945. — Vol. 26, no. 3. — P. 297–302. — https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.2307/1932409.

[9] Do Software Developers Understand Open Source Licenses? / Daniel A. Almeida, Gail C. Murphy, Greg Wilson, Mike Hoye // Pro-

ceedings of the 25th International Conference on Program Comprehension. — ICPC '17. — Buenos Aires, Argentina : IEEE Press, 2017. — P. 1–11. — Access mode: https://doi.org/10.1109/ICPC.2017.7.

[10] 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. — Access mode: https://doi.org/10.1109/SEW.2012.24.

[11] FOSSA official website. — 2020. — Режим доступа: https://fossa.com (дата обращения: 29.11.2020).

[12] FOSSology GitHub repository. — 2008. — Режим доступа: https://github.com/fossology/fossology (дата обращения: 29.11.2020).

[13] The GNU licenses. — Режим доступа: https://www.gnu.org/licenses/ (дата обращения: 29.11.2020).

[14] Gaudeul Alex. Do Open Source Developers Respond to Competition? The LATEX Case Study // Review of Network Economics. — 01 Jun. 2007. — Vol. 6, no. 2. — Access mode: https://www.degruyter.com/view/journals/rne/6/2/article-rne.2007.6.2.1119.xml.xml.

[15] 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. — Access mode: https://doi.org/10.1109/ICPC.2010.48.

[16] Gradle License Report GitHub repository. — 2020. — Режим доступа: https://github.com/jk1/Gradle-License-Report (дата обращения: 29.11.2020).

[17] Intellij Platform GitHub repository. — 2021. — Режим доступа: https://github.com/JetBrains/intellij-community (дата обращения: 29.03.2020).

[18] Intellij Platform SDK Docs. — 2021. — Режим доступа: https://plugins.jetbrains.com/docs/intellij/welcome.html (дата обращения: 29.03.2020).

[19] Java Research License. — 2020. — Режим доступа: https://www.oracle.com/technetwork/java/javase/jrl-5-150091.txt (дата обращения: 29.11.2020).

[20] KInference. — 2021. — Режим доступа: https://github.com/JetBrains-Research/kinference (дата обращения: 29.03.2020).

[21] The MIT License. — 1988. — Режим доступа: https://opensource.org/licenses/MIT (дата обращения: 29.11.2020).

[22] Markovtsev Vadim, Long Waren. Public git archive // Proceedings of the 15th International Conference on Mining Software Repositories - MSR '18. — 2018. — Access mode: http://dx.doi.org/10.1145/3196398.3196464.

[23] Maven Central dependency repository. — 2021. — Режим доступа: https://mvnrepository.com/repos/central/ (дата обращения: 29.03.2020).

[24] Open Neural Network Exchange. — 2021. — Режим доступа: https://onnx.ai/ (дата обращения: 29.03.2020).

[25] The Open Source Definition. — 2007. — Режим доступа: https://opensource.org/osd (дата обращения: 29.11.2020).

[26] Package Search Official Website. — 2021. — Режим доступа: https://package-search.jetbrains.com/ (дата обращения: 29.03.2020).

[27] SPDX License List. — 2020. — Режим доступа: https://spdx.org/licenses/ (дата обращения: 29.11.2020).

[28] Sourcerer's Apprentice and the study of code snippet migration / S. Romansky, C. Chen, B. Malhotra, A. Hindle // ArXiv. — 2018. — Vol. abs/1808.00106.

[29] 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. — ISBN: 9781450375177. — Access mode: https://doi.org/10.1145/3379597.3387455.

[30] Sunset Bintray JCenter Official Blog Post. — 2021. — Режим доступа: https://jfrog.com/blog/into-the-sunset-bintray-jcenter-gocenter-and-chartcenter/ (дата обращения: 29.03.2020).

[31] Swing — GUI widget toolkit for Java. — 2021. — Режим доступа: https://docs.oracle.com/javase/8/docs/technotes/guides/swing/index.html (дата обращения: 29.03.2020).

[32] Wikipedia contributors. Software license — Wikipedia, The Free Encyclopedia. — 2020. — [Online; accessed 14-December-2020]. Access mode: https://en.wikipedia.org/w/index.php?title=Software_license&oldid=991148912.

[33] 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. — Access mode: https://doi.org/10.1109/ICSM.2010.5609557.

[34] askalono GitHub repository. — 2018. — Режим доступа: https://github.com/jpeddicord/askalono (дата обращения: 29.11.2020).

[35] go-license-detector GitHub repository. — 2018. — Режим доступа: https://github.com/src-d/go-license-detector (дата обращения: 29.11.2020).

[36] gradle-license-plugin GitHub repository. — 2020. — Режим доступа: https://github.com/jaredsburrows/gradle-license-plugin (дата обращения: 29.11.2020).

[37] license-maven-plugin GitHub repository. — 2013. — Режим доступа: https://github.com/mycila/license-maven-plugin (дата обращения: 29.11.2020).

[38] licensechecker GitHub repository. — 2018. — Режим доступа: https://github.com/boyter/lc (дата обращения: 29.11.2020).

[39] licensee GitHub repository. — 2014. — Режим доступа: https://github.com/licensee/licensee (дата обращения: 29.11.2020).

[40] ootpa Troan Larry. Open Source from a Proprietary Perspective. — 2006. — Режим доступа: https://docplayer.net/5605674-Open-source-from-a-larry-ootpa-troan.html (дата обращения: 29.11.2020).

[41] scancode-toolkit GitHub repository. — 2015. — Режим доступа: https://github.com/nexB/scancode-toolkit (дата обращения: 29.11.2020).

[42] 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. 198b–198b.