Master's thesis

Master's Programme in Computer Science

Public Licenses in Software Engineering: A Multivocal Literature Review

Akira Taguchi

September 1, 2025

Faculty of Science

University of Helsinki

Contact information

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki,Finland

Email address: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Study programme
Faculty of Science	Master's Programme in Computer Science

Tekijä — Författare — Author

Akira Taguchi

Työn nimi — Arbetets titel — Title

Public Licenses in Software Engineering: A Multivocal Literature Review

Ohjaajat — Handledare — Supervisors

Prof. Tomi Männistö

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's thesis	September 1, 2025	38 pages, 18 appendix pages

Tiivistelmä — Referat — Abstract

Context: Public software licenses are central to the distribution of works in software engineering. For example in open source there must be an appropriate public software license attached to the source code in order for open-source software to be freely available for possible modification and redistribution. The large number of these public software licenses is one contributing factor in the difficulty of understanding how these licenses work.

Methods: As the goal is to explore and evaluate different public licenses in software engineering, whilst also taking a look into the different sites listing these public software licenses, this study adopts a systematic literature review approach. The search strings, search process and other relevant information are meticulously documented and explored in each step of the research process.

Results: 594 unique public software licenses were found from five different license listing sites. The found amount hints at the problem of too many public software licenses which might convert to the difficulty of understanding public software licenses, which again might lead to unexpected and unwanted legal agreements. Both research and industry have room for improvements such as new research using grey literature, use of `AGPL-3.0-or-later` or some other court-proven public software license and possibly supporting the new Post Open Source movement, that could revolutionaize the software industry.

Conclusions: Future research efforts should start at adoping a clear baseline including the terminology. Future industry efforts should focus on trying to learn and understand the practical effects of the public software licenses being used.

ACM Computing Classification System (CCS)
Social and professional topics → Computing / technology policy → Intellectual property → Licensing

Avainsanat — Nyckelord — Keywords

open source, free / libre software, copyright, proprietary software, copyleft, license, post open source

Säilytyspaikka — Förvaringsställe — Where deposited

Helsinki University Library

Muita tietoja — övriga uppgifter — Additional information

Software study track

Acknowledgements

ScanCode LicenseDB data is licensed under the Creative Commons Attribution License 4.0 (CC-BY-4.0) by nexB Inc. and others.

Thanks to Sami Nurmivaara for the thesis about green in software engineering. Mimicking the structure made it easy for me to write about my own subject within the area of software engineering.

Thanks to professor Tomi Männistö and professor Mika Mäntylä.

Thanks to Dr. Richard Matthew Stallman and Bruce Perens for helping with the thesis and the smaller projects related to it.

Thanks to Iikka Hauhio and Joonas Jakobson for giving a hint for using a license database. Thanks to Rashid Harvey and Barunes Padhy for sending me software licensing related videos and news that kept me challenged. Thanks to def for borrowing GPT-4 back when it was not a commodity.

Thanks to Suvi for supporting my work with the thesis from the beginning and enduring my mental health issues and being a good, caring mother.

Thanks to all my friends and family who were and are there for me.

Dedicated to Artemis.

Contents

1 Introduction

Public software licenses are central to the distribution of works in software engineering. In open source there must be an appropriate public software license attached to the source code in order for the piece of software to be freely available for possible modification and redistribution. Because open source is central to software engineering the licenses enabling open source should also be considered important in the same context.

The objective of this thesis is to have an understanding on how to retrieve all public licenses in software engineering by conducting a multivocal literature review (detailed goals are outlined in Sections 1.3 Thesis goal and contributions, 2.1 Research questions).

The review follows Kitchenham and Charter's guidelines for systematic literature reviews (Kitchenham and Charters, 2007). Furthermore, the review takes into account insights from practical experiences with systematic reviews (Mahdavi-Hezavehi et al., 2013)(Nurmivaara, 2023).

1.1 Background and terminology

The definition of a public license is a license by which a copyright holder as licensor can grant additional copyright permissions to any and all persons in the general public as licensees (Hietanen, 2007). By applying a public license to a work, provided that the licensees obey the terms and conditions of the license, copyright holders give permission for others to copy or change their work in ways that would otherwise infringe copyright law.

Understanding public software licenses can be difficult. This could stem from the legal nature of the license texts and the large number of already-existing public software licenses. The license texts usually favors correctedness over the readability for the developer. This is because the license text has to act as a valid legal instrument otherwise it cannot be endorsed (Ferguson, 2006). The lack of understanding of public software licenses leaves too much room for interpretation. In 2023 IBM's Red Hat seemingly violated the spirit of a popular public software license, the GNU General Public License version 2 (GPL-2.0) (McGrath, 2023; Kuhn, 2023). This was an unpleasant surprise to the public. If the

public software licenses would be more easily understood, the proprietarization of Red Hat Enterprise Linux (RHEL) would have been less of a surprise to the users. To give some context on the violation of the spirit of the GPL-2.0, the project behind GNU General Public License (GPL), GNU Project initially attempted to ensure the users via the GPL have to the following three freedoms (GNU, 1996):

- Freedom 1: The freedom to study how the program works, and change it so it does your computing as you wish. Access to the source code is a precondition for this.
- Freedom2: The freedom to redistribute copies so you can help others
- Freedom 3: The freedom to distribute copies of your modified versions to others. By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

On top of the legal details of public software licenses, based on the coverage of the incident we suspect that software engineers in general have a tough time understanding the pragmatic freedoms and restrictions of public licenses used in software engineering. In the instance of the RHEL incident it could have been even lesser of a surprise to software engineers if they would have known about other public software licenses and what they try to achieve or why GPL-2.0 has been succeeded by GNU General Public License version 3 (GPL-3.0).

The unestablished terminology presents challenges to discussing about the topic. In this thesis we will use the following definitions for the terms most commonly misdefined. Open source is defined as any software source code that is licensed under a public software license that the Open Source Initiative (OSI) has approved as ”open source”. Free software is software that has source code licensed under a public software license that the Free Software Foundation (FSF) has approved as ”free software”.

1.2 Background and terminology of public licenses

The current terminology is used inconsistently which leads to incorrectness in the field of software engineering. For example The Open Source Initiative (OSI) classifies GPL-3.0 under the term ”open source” whereas the Free Software Foundation (FSF) classifies GPL-3.0 under the term ”free software” (OSI, 2008)(Stallman, 2009). Some parts of the two definitions are mutually exclusive. This is rarely mentioned when people talk about Free

and Open Source Software (FOSS) or Free / Libre and Open Source Software (FLOSS) which leads to misunderstanding that the two approaches are the same. This is why our focus will be public licenses in software engineering, and not for example, FLOSS licenses in software engineering. This also distinguishes our investigation from the broader topic of copyright licenses or the copyright law. This also includes public software licenses that are not approved by the FSF nor OSI hence not falling under the group of FLOSS licenses. In this section we aim to increase the accessibility of our discussion by providing a concise overview of the background of the field of public software licenses and the terms we employ.

Another example of term inconsistency is the term "copyleft",which is defined by Mustonen (2003) in the following way:

> "Copyleft is a novel licensing scheme. It facilitates open and decentralized software development. Its key feature is that once a program is licensed by the inventor, the subsequent programs based on the original must also be licensed similarly."

Like with the definition of sustainability (Neumayer, 1999), copyleft also has the definitions of weak and strong within the term (Wikipedians, 2025). Weak copyleft licenses are often used to cover software libraries. This allows other software to link to the library and be redistributed without the requirement for the linking software to also be licensed under the same terms. Strong copyleft shares the same features Mustonen, 2003 presents regardless of the library nature of a piece of software. The general use of the term "copyleft" without the prefix also leads to inconsistency in the term usage.

To explain our emphasis on public licenses in software engineering, it is essential to examine the other possible areas of interest in public licenses. Our study classifies such efforts into eight domains as mentioned by the GNU Project (GNU, 2023). These domains include:

- public licenses in software engineering
- public licenses in documentation for example architecture documentation of a project that may or may not be software or even publicly licensed
- public licenses in artistic works for example digital art, music or videos
- public licenses in educational works
- public licenses in fonts

- public licenses in viewpoints
- public licenses in physical objects
- public licenses in other works

The primary aim of this study is to investigate public licenses in software engineering process. However, it is important to acknowledge that public licenses in software engineering is only one aspect of public licenses in general. These additional dimensions are crucial in adoption and implementation of public licenses in software engineering, but they are not the focus of this thesis.

For example, including artistic works such as music would require us to understand the basics of music theory and what sets apart distinct pieces of music from one another, something that could be outside the skillset of the author. While developing a comprehensive theory, framework, and tooling for public licenses as a whole is a gargantuan task beyond the scope of a single thesis, narrowing our focus to software engineering enables us to examine a more concise and complete aspect of the main topic of this thesis.

As significant point of clarification, it is essential to acknowledge that public licenses are generally meant to be used as valid legal instruments. The question whether or not a public license can act as a legal instrument is critical to the main function of these licenses. However, this thesis will not focus on the legal doctrine aspects either. The enforceability of public licenses has seen discussion in the academic field of law since the dawn of these licenses and since there's already an academic base for research it is likely the discussion seems to continue on with a healthy amount of activity (Duisburg, 2011).

Since the most recognized public licenses in software engineering are either open-source licenses or free software licenses and since both paradigms are driven by different organizations with different goals and values, it is understandable how non-standardized the terminology in the scope of public licenses in software engineering is. The example given in the first section of this sub-chapter illustrates the challenges involved in maintaining consistency in the use of terminology in this emerging field and further warrants a closer inspection of the terminology to emphasize our own standing in the field.

Let us explore further the differences and similarities between open source and free software at the software engineering level of public licenses. This is a crucial step since majority of public software licenses are either open source, free software or both. We glanced over the free software definition in the first section of Chapter 1. Open Source Initiative defines open-source licenses in the Open Source Definition briefly in the following way (OSI, 2024):

> "Open source licenses are licenses that comply with the Open Source Definition - in brief, they allow software to be freely used, modified, and shared."

Like the FSF with free software, OSI has the final word on what passes as open source and what does not. For example a new public software license will not classify as free software nor open source until the corresponding organization has acknowledged the public software license as either free software, open source or neither. If a public software license is accepted by both FSF and OSI it will fall under the term FLOSS. It is possible that a public software license gets accepted by neither of the organization or it gets rejected by both organizations. In general the strong copyleft free software license requirements are considered more strict than the open source license requirements. For the sake of perspective we could simplify the differences like so: free software requires the redistributions of the licensed software to be open as well but open source licenses do not usually require this. The terms free software and open source are in general often misunderstood or just thought of as FLOSS collectively because the terms have a hard time conveying their paradigms in the natural language. One would not think free software does not mean software free of charge nor would one think that open source allows closed source redistributions of the licensed software. We will glance over the impacts on the industry of these two terms in Chapter 4.

With the context laid out in this chapter let us define public licenses in software engineering for the purpose of this study: Public software licenses are copyright licenses where the licensees are not limited and the copyright license in question is meant be used in licensing software source code. This helps us create the search strings and find the relevant literature for this thesis. This also helps us exclude public licenses regarding documentation, media and all other non-software targeted public licenses.

1.3 Thesis goal and contributions

The quest to categorize every public software license under some paradigm objectively is a complex one and cannot be comprehensively answered in a single paragraph. Therefore it is essential to continue taking the correct steps towards incresing the scientific understanding and providing the industry with examples, standards and processes to follow. However, as the following chapters reveal, a significant amount of effort is still being spent on solving the same problem multiple times, rather than building on existing knowledge and finding the next problem to solve. This thesis aims to contribute to mitigating this challenge

by providing a rigorous analysis of the current state of the field. As the knowledge, conventions, and terminology take shape,we can look forward to reaching a state where less effort is spent on defining concepts and more on practical problem-solving.

This study has two main viewpoints. The first one is to provide rigorous multivocal research on public software licenses to the academic field. Because this thesis already does the multivocal work on public licenses in software engineering, the researches of the future can cite the results of this thesis without having to mark their study a multivocal one. This is the grand goal of this thesis. The second one is to provide insights and general metrics to the professional field of software engineering on public software licenses. Hopefully this makes conversation on public licenses in software engineering easier and more rooted to scientific research rather than gut feeling and old, non-scientific articles on the insights and metrics of public licenses in software engineering.

The primary goal is to have an understanding on how to retrieve all public licenses in software engineering by conducting a multivocal literature review. The research aims to provide a novel perspective on relevant licenses and to extract key findings through a rigorous literature review process.

This thesis' academic and industrial contribution is an attempt to solve these problems in a structured manner. In Chapter 1 we state definitions and terminology used in the scope of this thesis. We go over the reasons why there does not exist consistent terminology in this area and conversely why the definitions are the most stabile ones in this area. In chapters 2 and 3 we take a deep dive into the public software licenses through a multivocal literature review. In chapters 4 and 5 We include our own suggestions and basic knowledge for professionals and academics in the industry to enhance the understanding of public licenses in software engineering. This step also includes discussion of the future research and contributes to stablizing the terminology and reinforcing the already-existing definitions in the academic field.

1.4 Thesis structure

This thesis follows the IMRaD structure. Chapter 1 introduces the problem, this thesis' possible contributions and some further background. Chapter 2 goes over the process and the methods of the multivocal literature review. This is where most of the actual research takes place in. Chapter 3 presents results to the research questions. Chapter 4 discusses implications for research. The chapter also discusses software engineering professionals in

the thesis' context and the validity of the thesis' research. Chapter 5 concludes this thesis with the help of the research questions and the future of the research.

2 Methods

This chapter aims to establish a precisely defined and rigorous research approach to enhance transparency and repeatability. We will take the steps required to ensure that every phase and decision is thoroughly documented, enabling the reader to retrace the research process. In a thesis made by a single researcher, the lack of cross-examination of results with multiple researchers and the validation of evaluation criteria for opinion bias pose threats to validity, as will be clarified further in Chapter 4. Therefore, special attention will be paid to address these concerns. By following this approach, this research endeavors to contribute to the existing body of knowledge in the field of computer science in a robust and reliable manner.

The systematic literature review method is a well-established approach for conducting a comprehensive and rigorous analysis of the existing research on specific research question or subject (Kitchenham and Charters, 2007). This paper presents a multivocal literature review. Multivocal literature review is type of systematic literature review that includes both academic literature and grey literature (Garousi et al., 2019). This method was selected for this study to facilitate a thorough and scientifically interdisciplinary examination of public licenses in software engineering. The existing literature consists of public software licenses not found in academic databases and as such are considered gray literature, making the thesis a multivocal literature review.

This study follows the guidelines outlined by Kitchenham and Charters, 2007, to ensure its quality. The multivocal review method consists of three distinct phases: planning, conducting and reporting the review. This study stricly adhered to this structure. The phases can be further broken down into a research protocol, as illustrated in Figure 2.1. Adhering to the protocol is the first step in ensuring a well-documented and rigorous process, which increases the validity and auditability of the study.

The multivocal literature review process began with the formulation of research questions and the establishment of a comprehensive search strategy and scope. The search process was conducted by employing a quasi-gold standard (QGS) approach based on the implementation by Zhang and Ali Babar, 2010. After the completion of the search process, the inclusion and exclusion criteria were defined. To ensure a structured evaluation of the literature, a data extraction form was created. Finally, a strategy for analyzing the

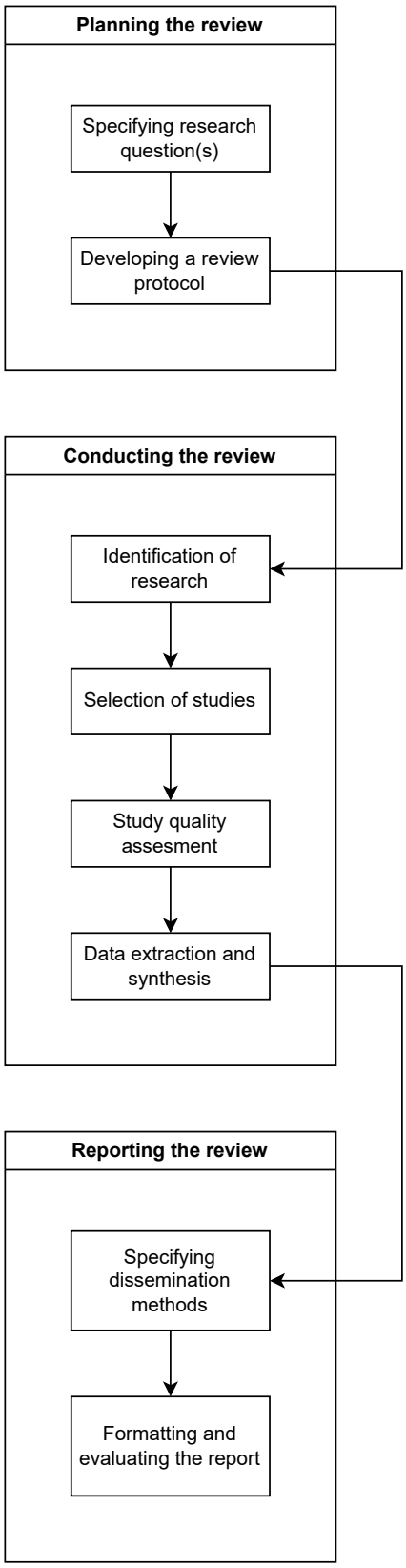

Figure 2.1: Three phases of a systematic literature review

extracted data from the literature was designed.

To ensure the reliability and validity of the research protocol, it was validated against similar systematic literature reviews in computer science, the aforementioned guidelines by Kitchenham and Charters, 2007, and was further refined through an iterative process. Specifically, a subset of the data was tested on (The QGS) and any identified issues or problems were recorded and addressed. The details of this process are explained and thoroughly documented in the following sections. Similarly, the same approach was followed for the data extraction process, whereby a subset of literature was tested to refine the data extraction form. The revision of the form was undertaken as necessary to guarantee the completeness and accuracy of the extracted data.

2.1 Research questions

The research questions in this study served two primary purposes. Firstly, they aimed to provide an analysis of the existing multivocal literature on public licenses in software engineering for the researchers interested about the field. Secondly, the questions were designed to cater a secondary audience of professional software engineering practicioners. As discussed in the Chapter 1, the following research questions were addressed in this thesis:

- RQ1: How many public licenses in software engineering does there exist?
- RQ2: How consistent are the naming conventions for public licenses in software engineering
-

The multivocal literature review in this thesis begins with addressing RQ1, which aims to provide the amount of public software licenses that exist in our five public license listing sites in total. This information could be most valuable to the researchers. The results can be used to introduce some notable background of the current public licenses in software engineering and enabling focus to more specific areas inside the topic of this thesis.

Next RQ2 seeks to find the amount of duplicate licenses between the license listing sites. Results to this research question are also mostly useful to researchers of the field. Moreover, the documented methods are most likely the most valuable information for the researchers.

Finally RQ3 attempts to count the total number of individual public software licenses within the scope of this thesis. The research question builds on top of the results and methods of the previous research questions. This information could be most valuable for the practitioners since it could give some overview and a sense of the scale when picking a public software license that would serve the practitioners' needs the best.

2.2 Search strategy

The search process was conducted on five public license listing websites. The selection criteria for the literature were defined after the search process and the selection process was based on inclusion and exclusion criteria. The inclusion and exclusion criteria and each step of exclusion on the literature found are presented later in this chapter. Originally the search terms would have been applied to the license listing sites directly just like in a normal multivocal literature review or in a systematic literature review. Keywords however produced highly varying and non-reproducible results in Google Scholar and Google Search. Some license listing websites such as FSF's list of pages categorized as licenses could not be found from Google Search even with the `site` operator: `site:https://directory.fsf.org/wiki/Category:License`. Although the page has been up since 2013, for some reason Google has not crawled the page in 10 years (FSF, 2024). This is why this thesis does not include search terms of the initial phase per se but rather inclusion and exclusion strings on the second phase. For the sake of validity the thesis still follows the guidelines presented in Kitchenham and Charters, 2007 with the exception of replacing an academic database search engine with the five license listing sites, web scraping and our own Python script performing the legwork of an academic database search engine in a systematic literature review.

The data extraction process was performed in a standardized and systematic manner, with the aim of obtaining the relevant information from the selected literature. The data extraction form used included license shortcode used in the listing site, listing site name, full license text and is available Table 2.2. The extracted data was then used to answer the research questions and perform the data analysis. The results of the data analysis were then reported in a rigorous manner.

2.2.1 Search method

The search was conducted on five license listing websites, as mentioned earlier, to obtain a broad set of multivocal literature. This approach yielded a large number of literature that were processed to a subset of high-relevance literature using inclusion and exclusion criteria presented later in this chapter. Manual searching of databases with hundreds of public licenses is not feasible, and it is prone to researcher bias and may overlook relevant venues from other scientific disciplines. However, a preliminary manual search was performed to reduce the number of iterations required and establish the quasi-gold standard (QGS) mentioned earlier.

2.2.2 Search scope and terms

The search terms, or in our case, the inclusion and exclusion string was determined through an iterative process that took into account the research questions and topic. Synonyms for key terms were included and combined using Boolean logic to form a comprehensive inclusion and exclusion string. As mentioned earlier the inclusion and exclusion criteria are presented later in this chapter.

The inclusion and exclusion string was established on a basis of quasi-gold standard as proposed by Zhang and Ali Babar, 2010. For establishing a quasi-gold standard we employed a manually crafted inclusion and exclusion string based on the topic and research questions of this study. As we defined public licenses in software engineering as licenses where the licensees are not limited and the license in question is meant be used in licensing software source code in Chapter 1 and our research questions focus on finding useful metrics about the public licenses, we manually formulated the inclusion and exclusion string in Python:

```
^(?!.*\b(documentation\s+license|creative\s+commons|open data)\b).*
```

In order to run the inclusion and exclusion string that established the quality-gold standard against the literature we had to gather them first. We started defining our search scope from the Wikipedia page of one of the most used open source license (Balter, 2015), the MIT license (Wikipedians, 2024). The infobox contained fields in the order shown in Table 2.1.

The validity threats regarding this choice are discussed in a later chapter. The publisher, GPL compatibility, copyleft and the linking exception did not result in any meaningful

Field	Value
Publisher	Massachusetts Insitute of Technology
SPDX identifier	MIT
Debian FSG compatible	Yes
FSF approved	Yes
OSI approved	Yes
GPL compatible	Yes
Copyleft	No
Linking from code with a different license	Yes

Table 2.1: MIT License Wikipedia page infobox

URL
https://spdx.org/licenses/
https://wiki.debian.org/DFSGLicenses
https://directory.fsf.org/wiki?title=Category:License
https://opensource.org/licenses
https://www.gnu.org/licenses/license-list.html

Table 2.2: License listing sites chosen

license listing websites. This leaves us with the SPDX, Debian FSG compatibility, FSF and OSI from which all resulted in some sort of license listing websites. Since the fields were roughly as follows: SPDX, FSF, OSI and GNU, after some investigating, we decided to start the search for public software licenses from the following license listing sites:

The web pages were scraped of the public license shortcodes using the browser's developer tools. These shortcodes were imported into a spreadsheet editor with each shortcode under their corresponding listing site name. This resulted in 1057 public licenses. Because the same public license would sometimes occur in multiple listing sites strictly duplicate shortcodes were removed using the spreadsheet editor resulting in 780 public licenses. Removing the duplicates was not intelligent and left duplicates like ZPL-2.0 and ZPL - 2.0 as unique license shortcodes. The solution to this problem is presented in a later sub-chapter. The table in the state after the strict removal of duplicates is provided in this thesis' repository (Taguchi, 2025) under the name of `stage1-licenses.md`.

With the search for the initial license listing websites completed we moved onto the search

process itself.

2.3 Search process

The literature selection process was divided into multiple stages, as outlined in Figure 2.2. The initial step involved the formation of a inclusion and exclusion string through the use of a quasi-gold standard.

In the first stage, the search was conducted using the web pages titled ”MIT License” (Wikipedians, 2024), ”SPDX License List” (Linux Foundation, 2024), ”The DFSG and Software Licenses” (Debian, 2024), FSF’s ”Category:License” Wiki page (FSF, 2024), GNU’s ”Various Licenses and Comments about Them” (GNU, 2023) and "OSI Approved licenses" (OSI, 2024) focusing, focusing on the license listing site name and shortcode. Then we identified and eliminated any duplicates, producing a preliminary set of potentially relevant literature. The dataset after the first stage of the search process is provided in this thesis’ repository (Taguchi, 2025) under the name of `stage1-licenses.md`, as mentioned earlier.

In the second stage, the inclusion and exclusion criteria were applied to further filter the literature and reduce the number of licenses to be reviewed. Then the resulting dataset was taken for a closer look where the quality of the literature was examined manually by the author, resulting in some manual exclusions based on the content and availability of the literature. This is where the inclusion and exclusion string took concrete place in which means the results were cross-referenced with the quasi-gold standard to validate it. The verbal reasoning for manual exclusions is clarified in Sub-chapter 2.4. The dataset after the second stage is provided in this thesis’ repository (Taguchi, 2025) under the name of `stage2-licenses.md`.

The third stage was the most time-consuming and involved a manual review of the full license texts. After reading and evaluating each license, a final round of exclusions was completed and documented. The remaining licenses were used for data collection and analysis in the final part of the study. The final list of licenses is available in Appendix A.

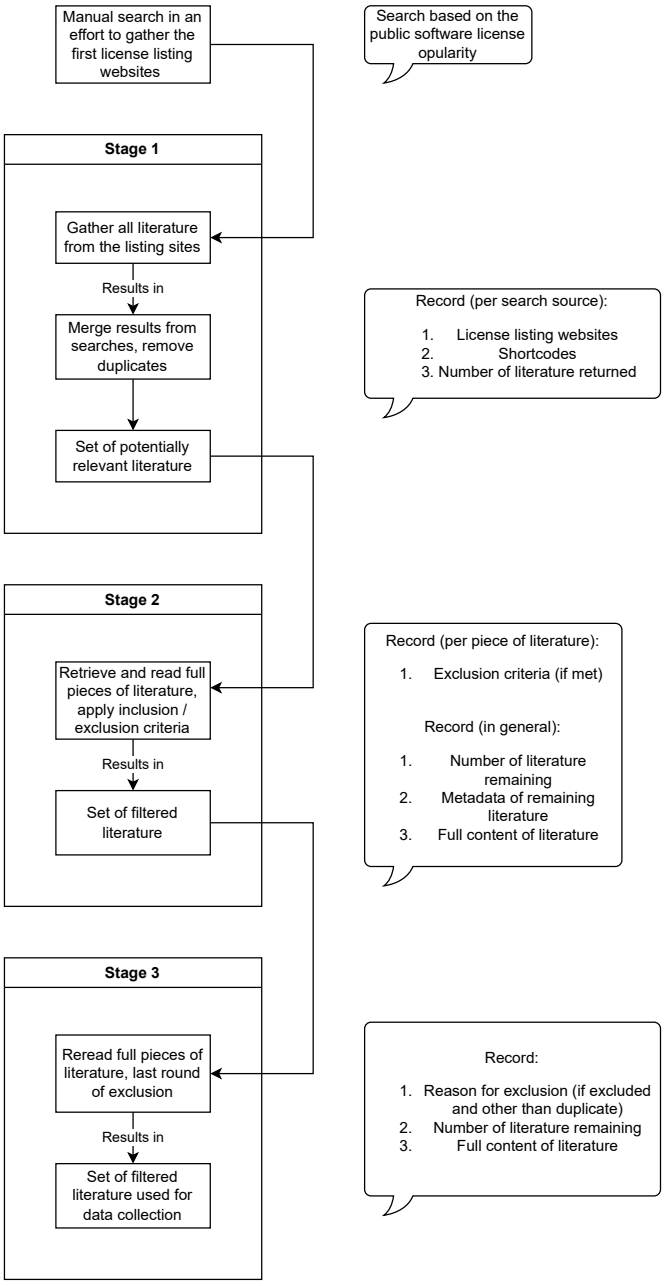

Figure 2.2: Search process divided into stages

2.4 Inclusion and exclusion criteria

Before we could apply the inclusion and exclusion criteria to the literature we had to fetch the full license texts from somewhere since the first stage's dataset included only the shortcode and listing site name. We decided that the public license database by ScanCode published in GitHub (ScanCode, 2025) was to be used fetching the initial full license texts based on the shortcodes of the first stage. The license database could be found by searching GitHub with the term "license database". The monolithic Python script used for this matching and fetching is provided in this thesis' repository (Taguchi, 2025) under the name of `methods.py`. Some shortcodes from the first stage did not match any full license texts from the license database. We had to manually fetch the missing full license texts from these license listing sites. The fetching was done systematically in cycles until no more missing full license texts were found with the help of the first stage spreadsheet. as can be seen from the Python script. We now had a complete Python dictionary with the shortcode as the key and full license text as the value.

To be eligible for the data collection and analysis, a license had to match the following inclusion and exclusion regular expression string:

```
^(?!.*\b(documentation\s+license|creative\s+commons|open data)\b).*
```

The regular expression string first included some inclusion matching but we soon realized it would be more efficient to exclude licenses than to include licenses. To establish the quasi-gold standard we first tried to exclude all full license texts including the words "creative commons" since we knew the Creative Commons licenses are not suitable for computer code, which is our scope. We then opened all excluded licenses into tabs in our text editor which the Python script assigned to their respective directory of excluded licenses based on the regular expression. We then looked quickly at the first few lines of the full license texts and judged if the license was indeed not suitable for our scope. Then we did the same to the included licenses in their respective directory and glanced if there were some types of licenses that were not suitable for our scope. We did one more cycle like this to finally exclude the string "documentation license" as well. During the two cycles we manually marked some licenses as included or excluded regardless of the regular expression matching since there did exist some corner cases the matching missed. This ended the second stage of search process which focused on inclusion and exclusion criteria.

#	Field	Concern/Research question
F1	Shortcode	RQ2, RQ3
F2	Listing site name	RQ1, RQ2
F3	Full text	Documentation

Table 2.3: Data extraction form

2.5 Data collection and data analysis

To answer the research questions of this thesis, a thorough examination of the selected primary literature was conducted and the necessary data was collected using data extraction form presented in Table 2.2. A record of the full license texts after the third stage was kept for anaysis and is available as a directory called `stage3-licenses` in the thesis' repository (Taguchi, 2025).

To get the necessary from the remaining literature we decided that the next reasonable step was to remove duplicates from the licenses as systematically as possible. We used Python's `difflib` library to sort the full license texts. The library itself used the Ratcliff and Obershelp algorithm compare every full license text to every full license text. The time complexity is $O(n^3)$ and $\theta(n^2)$. This took our working computer 68 minutes each time the algorithm ran so we decided to only do one cycle of this type of duplicate removal. The licenses were outputted to `duplicate-finding` directory with the naming convention of `n-shortcode.txt` where the n was the sort order given by the Ratcliff and Obershelp algorithm. We then opened these licenses to tabs in our text editor and compared the full license texts by human eyes if they were actually the same license with little to no noise difference in the full license texts. The comparison tool of our text editor was also a helpful automation tool for longer licenses that could not fit into one computer display as whole without scrolling. "open data" was also applied in the regular expression string at this stage. While this should have happened in stage 2, we wanted to be honest that this exclusion really did happen in stage 3. At this stage we also manually excluded individual licenses from the final dataset if there was too much noise. For example, GNU listed licenses often as whitespaces or something alike. Examples of these will be given in Chapter 3. This ended the third stage of the search process and provided us with the necessary data to answer the research questions.

The subsequent chapter presents the outcomes of the steps taken in the study, as discussed

above.

3 Results

This chapter employes the data extracted from the set of primary literature, available in this thesis' repository (Taguchi, 2025) under the name of `stage1-licenses.md`, utilizing the methods outlined in Chapter 2 to address the research questions. Firstly, a summary of the general statistics collected and aggregated from the studies is presented. Following that, an analysis of the data is performed to provide answers to each of the research questions.

To begin with, the publication year was not limited and could not have been limited in a rigorous way. Almost all of the public software licenses came from different sources although they were listed in the five license listing sites. To give a rough estimate, one of the earliest public software license aiming for legal compliance was the original GPL from 1989 (Bernelin, 2020). The search was carried out by web scraping all of the licenses from the five license listing sites without any filters to the attributes of the licenses. The initial search results included 1057 public licenses, but after the exclusion and quality criteria of software-only license scope, the final resulting dataset was reached.

Given the large starting dataset, a simple statistical overview of the literature was generated and is presented in Figure 3.1 and Figure 3.2 with the full list of literature available in this thesis' repository (Taguchi, 2025) under the name of `stage1-licenses.md`. The statistics highlight some immediate observations, such as the volume differences between the five sites and how the initial volume doesn't correlate to the amount of duplicate one site holds compared to the four others.

After establishing the quasi-gold standard and completing the preliminary study review outlined in Chapter 2, we systematically searched for relevant literature using the five license listing sites. The resulting search findings were filtered through a set of inclusion/exclusion criteria, followed by an extensive evaluation of quality before the final step of manual review. The final collection of literature consisted of 594 licenses, for which we obtained and reviewed the complete texts while completing the data extraction form as presented in Table 2.1. To enhance transparency of the process, Figure 3.3 illustrates the progression of literature through each stage. We observed the number of literature acquired is adequate to gain an representative overview of the field, which we will explore further in this chapter.

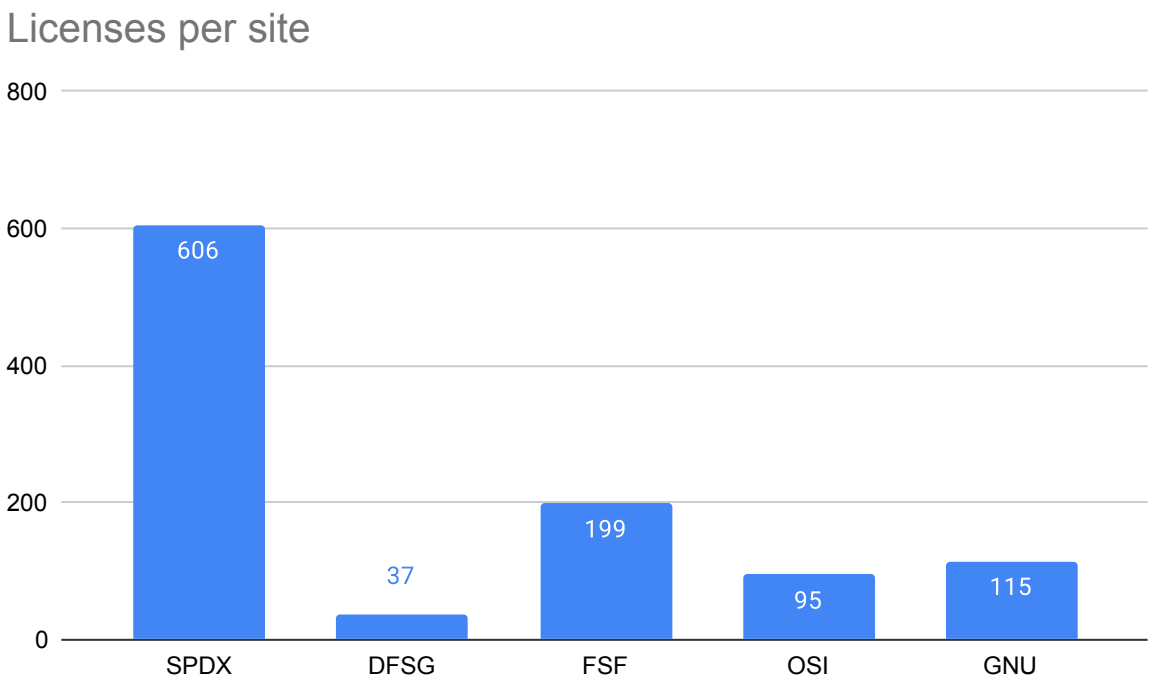

Figure 3.1: Statistics about the original 1057 licenses

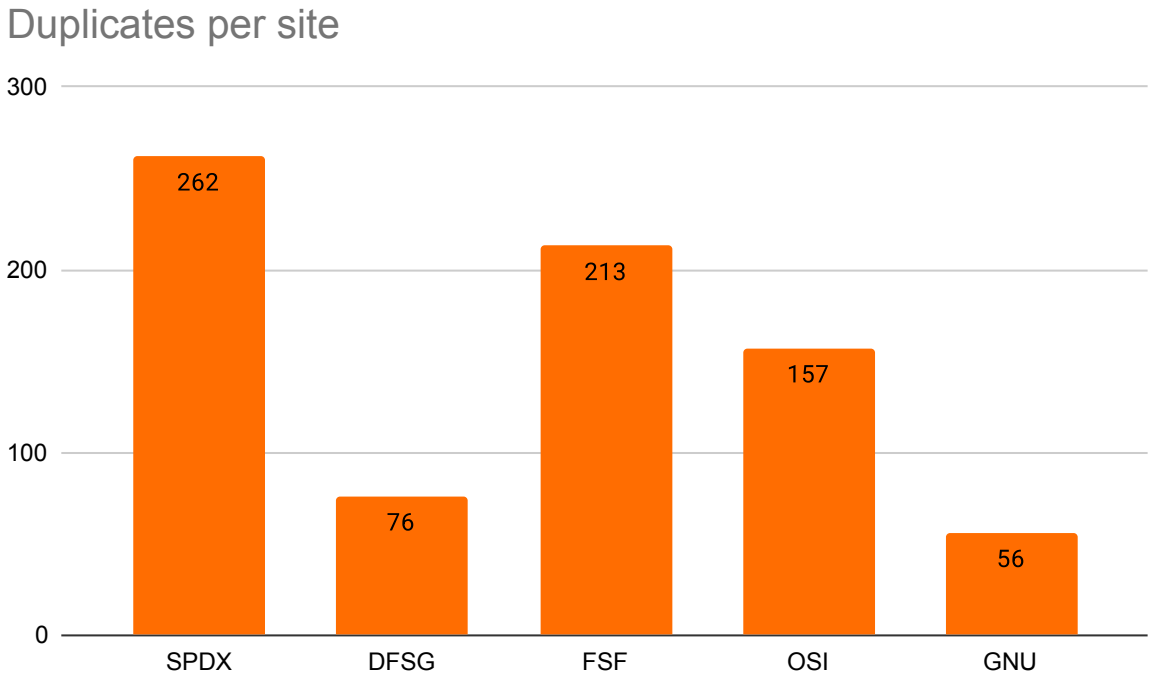

Figure 3.2: Duplicate statistics about the original 1057 licenses

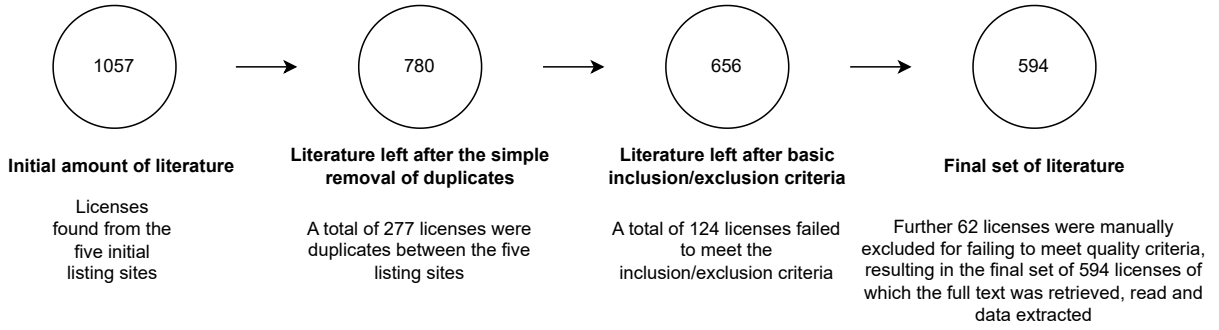

Figure 3.3: Number of literature per step of data collection

After this overview let us take a look into the specific research questions and their answers.

3.1 Five license listing sites and their licenses (RQ1)

Only license listing site during the first phase of the search process that did not give too much trouble was the SPDX. This was mostly be due to the table format used in that particular license listing site, shortcode identifier being provided inside that table and that it seems that SPDX preserves many of the licenses regardless of legal validity, superseding licenses or voluntary retirement.

The four other listing sites were more problematic to scrape. An example of DFSG is that for example for the license ”Licence Art Libre (Free Art License)” I had to just use the more commonly used shortcode `FAL` because the shortcode itself was not listed in DFSG. FSF used a Wiki as the base for the license listing site so there was plenty of missing and outdated data. For example the original `Python` license has the full text of just ”test” which indicates a pure mistake or a placeholder from the FSF. OSI had licenses that were retired between the search stages one and two. For example `cvw` was listed during the initial web scraping of the five listing sites but during the full license text fetching from the public license database of ScanCode the original creator of `cvw`, MITRE had voluntarily retired their license from the OSI license listing site and was no longer found from the OSI license listing site even though the missing license was noted to be from the OSI in the spreadsheet. The license was found however under the shortcode `cvwl` from the ScanCode public license database. This specific issue could have been solved by using an internet archiver, which we will discuss further in Chapter 4. GNU for example listed licenses like `attpubliclicense` which pointed the full license text to reside in and FSF site but the full license text was empty. In this case we had to just use the comments

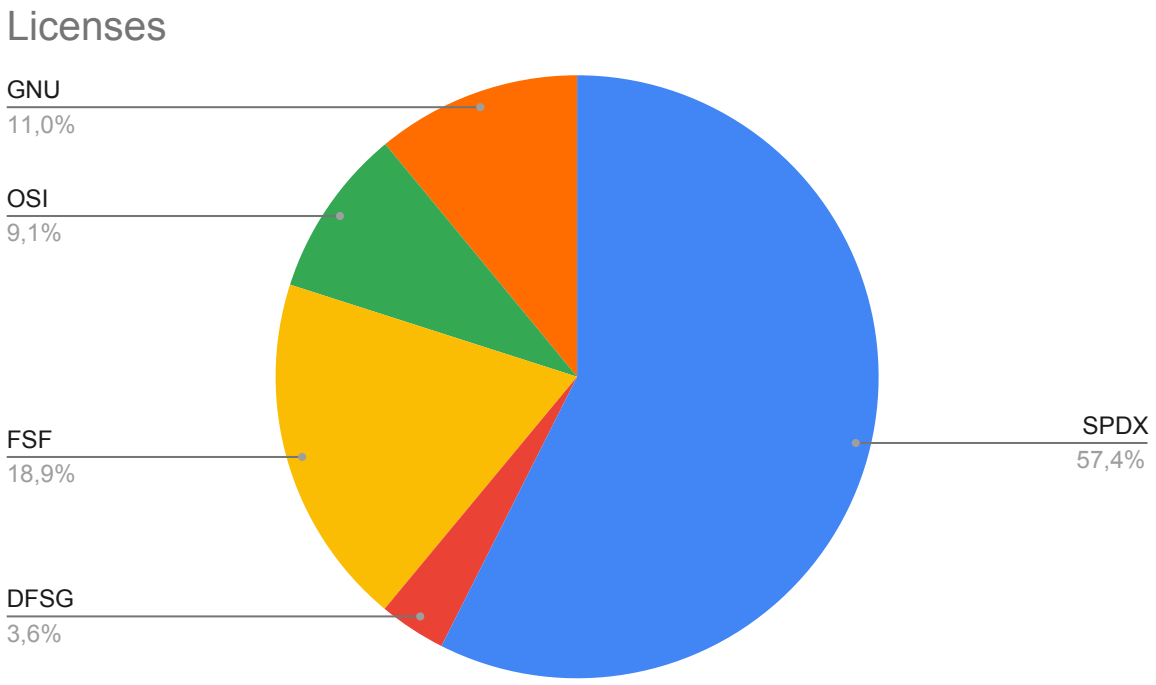

Figure 3.4: Distribution of license across the listing sites

made by GNU as the full license text in this thesis.

These are just individual examples of the level of maintenance appearing in the four other license listing sites and some other examples will be brought up later in the next chapter. This is because solving these problems also becomes more of a validity threat of the author regardless of the fact that this kind observation is a relevant result on its own in this thesis. When we count the amount of license per listing site after search stage one we get 607 licenses from the SPDX, 38 from the DFSG, 200 from the FSF, 96 from the OSI and 116 from GNU. The distribution of licenses from the five listing sites is illustrated in Figure 3.4.

3.2 Duplicates in license listing sites (RQ2)

A total of 277 duplicates compared by exact shortcode after search stage one and 62 more duplicates were found after applying the exclusion criteria after search stage three. The intial 277 duplicates indicates some level of disagreement about the shortcodes between the five license listing sites. This is because these shortcodes are not treated unique across the listing sites. The following citation from GNU (GNU, 2023) about the `MIT` license

demonstrates clearly the phenomenon affecting all five listing sites:

> ”Some people call this license ’the MIT License,’ but that term is misleading, since MIT has used many licenses for software. It is also ambiguous, since the same people also call the X11 license ’the MIT License,’ failing to distinguish them. We recommend not using the term ’MIT License.’
>
> The difference between the X11 license and the Expat license is that the X11 license contains an extra paragraph about using the X Consortium’s name. It is not a big deal, but it is a real difference.”

GNU calls the `MIT` license as the `Expat` license. A notable observation being here that the `MIT` license was used in 44.69% of repositories in 2015 (Balter, 2015) and with 4.8 million unique Git pushers in 2024 with the `MIT` license used as the license for the source code making the `MIT` license the most used license in GitHub in all four quarters of 2024 (GitHub, 2025). The same license was also used the most between the years 2020 and 2023 as can be seen in Figure 3.5. GNU demonstrates the same phemonemon of shortcode disagreement with the `GPL-3.0` license being either `GPL-3.0-only` or `GPL-3.0-or-later`, disagreeing with any other shortcode staten by the other four listing sites (GNU, 2023). The 62 duplicates after search stage three fortifies our result that the unique shortcodes do not provide unique full license text between the license listing sites. We conclude that the FSF, DSFG, SPDX and the OSI share the same problem of disagreement between some license shortcodes.

3.3 Total amount of public software licenses (RQ3)

A notable observation regarding the total amount of existing public software licenses is that many of the licenses might not be considered legally valid in any court and many of them do not even try to be legally valid in the court. An example of the former, possibly not court-fireproof software license is the `MIT`. The breach of the license is not meant to be settled in court but rather just agreeing developer to developer that the distributed software contains MIT -licensed source code. Some examples of the latter include `Beerware` and `JSON` from which the former recommends buying a licensor a cold beverage and the latter forbids the use of software to evil purposes.

The total amount of public software licenses after search stage three was 594. As mentioned before, in the last stage we excluded manually some licenses and removed some duplicates

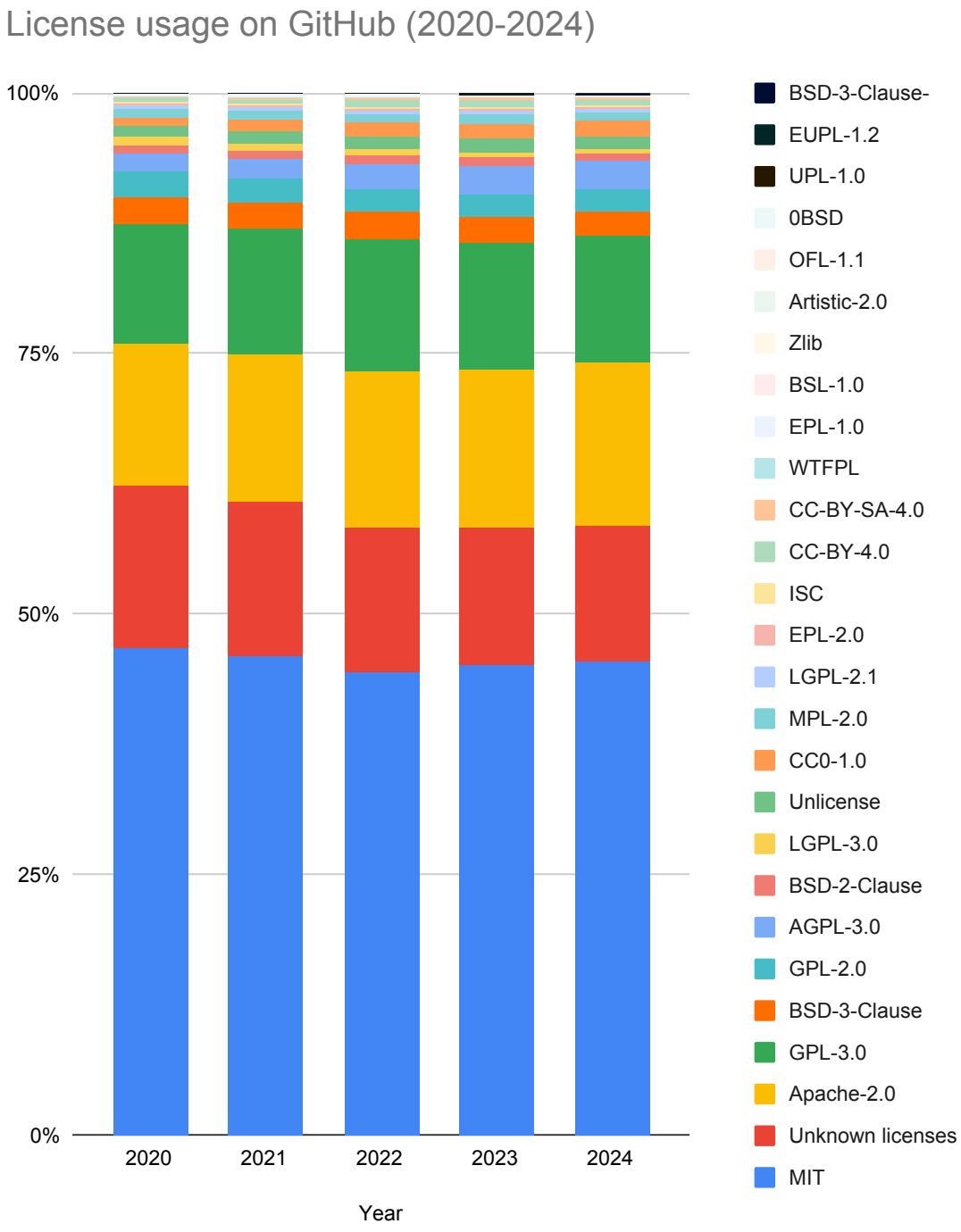

Figure 3.5: License usage between years 2020 and 2024 in GitHub

found using the Ratcliff and Obershelp similarity finding algorithm. The former included the original `Python` license, a Creative Commons license that did not have the words "creative commons" inside the full text and a Japanese Creative Commons license, which also did not have those words in English in the license full text. The 87 excluded licenses were also reviewed by their full license text and two exceptions were included manually. Both licenses were `CAL` licenses that are themselves licensed under a Creative Commons license so the exclusion filter gives us a false negative on these two licenses. We believe the amount of unique public software licenses found, the record of the full license texts and their shortcodes available as a directory called `stage3-licenses` in the thesis' repository (Taguchi, 2025) and the documented, systematic approach to finding answers to our three research questions, are the most notable results of this thesis.

4 Discussion

This research indicates that the field of public licenses in software engineering, meaning public software licenses, is at an early stage of development. The review of existing literature reveals a notable absence of common set of definitions and terminology in this field. Consequently, this often leads to work covering the same ground. Furthermore, the variability of terminology across different literature makes it challenging to compare and synthesize results effectively. Addressing this challenge will require the development of common measurement tools and frameworks for evaluating and comparing public licenses in software engineering. Such efforts could lead to the establishment of widely adopted standards for measuring and improving public software licenses.

That said, there is clear interest in public licenses in software engineering. Although the amount of purely academic literature on public software licenses is limited, the amount of grey literature on public software licenses is ever increasing. This is most likely due to the recent developments in the field regarding proprietarization noted in the Chapter 1.

A notable observation of our reserach is that the recent efforts in the industry have led to the development of Post Open Source not yet noted in the research literature reviewed (Claburn, 2023). Whether or not this paradigm will change the industry like open source and free software did will only be evident over time. The paradigm is explained shortly in Chapter 5.

4.1 Implications for research

To improve the maturity of research methods employed in the field of public licenses in software engineering, researchers should aim to use more rigorous and comprehensive research methods. This may involve using larger and more diverse data sets, developing more sophisticated measurement tools, and conducting experiments that are representative of real-world scenarios.

Furthermore, researchers should strive to increase the transparency and reproducibility of their research by making their data and code openly available. This would enable other researchers to replicate and build upon their work, as well as facilitate the establishment

of common standards and best practices.

Finally, it is important for researchers to publish more articles regardless of the grey literature included in the papers. Because there is largely only grey literature published in the twenty-first century in the field, the next academic articles will be multivocal by default. The non-multivocal, academic articles will follow but only after there are systematic, academic and multivocal articles published for the former to build on. The results presented here are modest but by working together, researchers and industry professionals could produce more useful research regarding public licenses in software engineering.

4.2 Implications for software engineering professionals

Software engineering professionals should start by educating themselves of the basics of public licenses in software engineering and incorporating it into their design and development processes. They should be mindful or strive to mindful about the public licenses their third-party softwares are using and how it impacts their craft. Making a map of the public software licenses and their corresponding usecases might help plotting the larger picture.

However, it is important to acknowledge that the institutions should hold the greater responsibility of teaching the basics of public software licenses without getting too tangled up in history, politics or simply waving the field of as a form of human rights activism. The key focus points here being vocational schools, software engineering courses, college and university since these are the timestamps where most software engineers start to produce code that need to be licensed or require the use of a licensed piece of software.

Overall, the lack of public software license knowledge regarding software engineering professionals points to the need of more education regarding public software licenses and the practical effects stemming from the application of these licenses.

4.3 Limitations and threats to validity

The major limitation of this study is that the subjective results could not be validated by multiple researchers. In a systematic review, it is standard practice and highly recommended to have at least two, if not more, individuals independently conduct the review

processes and then cross validating the findings. This would result in the possibility of comparing individual exclusion decisions and other decicions, thereby increasing the credibility of the study. However, in this study, the methodology was thoroughly documented, which allows us to assert with confidence that the study has an appropriate level of of validity.

As a work of single researcher, there is also a chance of inaccuracy and bias in the literature selection and filtering process. As much of the literature had to be reviewed manually and then included/excluded on a qualitative basis, this is a known limitation and a threat to validity. Multiple rounds of documented filtering and a clear paper trail of all decisions made keeps this threat in the acceptable levels.

4.3.1 Limitations of literature selection for review

Efforts were made to ensure the inclusion of comprehensive set of literature in the search process. This was achieved by setting the starting point of license listing sites to the Wikipedia article of the MIT license.

The first phase of filtering has some notable limitations starting with the two license listing websites: SPDX and DFSG. Since the material was gathered to a spreadsheet program the duplicates were removed using the short identifier the listing page was using. Let's look at this validity threat using an example. Suppose our spreadsheet program has acquired the public license with an identifier "MIT". The results of phase one will not include any other public license marked with the identifier "MIT". In the worst case the identifier "MIT" could have actually been "MIT-DFSG-edition" but with the identifier of "MIT". Since there were so many public software licenses in phase one it would not have been possible to check the uniqueness of all removed duplicates. One of the reasons why this would not have been feasible is that the listing sites would fetch the public license contents from another webpage or at the second worst case, from another website. The worst case is that the URl is dead and we get HTTP 404. The amount of public software licenses, duplicates and the lack of already existing tools makes this problem multilayered. However this is the level of integrity we decided to finish our study with.

FSF's license listing introduced us to pick another limitation for the scope of this thesis. The license shortcoded as "other" was not a public license but instead a hyperlink to another listing webpage that listed programs that the FSF has no yet managed to document the license which the program uses. Although the one of the programs called "babl" was

licensed as with ”gplv3” the amount of undocumented programs was over 5200 at the time of observation. For this reason we are excluding the public software licenses found indirectly from the category ”other”.

GNU project’s listing site allowed us to use a shortcut of sorts which we will document here for the purposes of acknowleding the limitations of it. The table of contents at the listing site marked certain consequtive public software licenses as software public software licenses. On top of this the public software licenses were not organized into easily processable tables but rather in stacked on one another in rich text format. Although we decided to use regex on the HTML file the included public software licenses were only the ones that were simply under the header ”Software licenses”. In the worst case scenario GNU project could have misinterpreted some public software licenses as non-software licenses thus making this thesis exclude them with a wrong reason. While from a quick glance and the existence of the other four license listing sites, we think it is still worth documenting when it comes to validity and the integrity of this thesis.

On top of too heavy filters we would also like to document the too light filters in the literature selection for review. We can see from Appendix A that for example public software licenses with the literature identifiers L777 and L780 are almost the same regarding the shortcoded identifiers: ”ZPL - 2.1” and ”ZPL-2.1”. The duplicate removal would have been seemingly simple to execute on phase 1. However with the presence of over 700 pieces of literature we decided not to give special treatment to any potential set of duplicates. While it is most possible that OSI’s ”ZPL - 2.1” is equivalent exactly to SPDX’s ”ZPL-2.1” we could not be sure without looking at their contents. This could have resulted duplicate public software licenses in the literature selection for review but these type of duplicates are removed in phases 2 and 3 due to the public software licenses being read in full.

To finish this subsection we will discuss some more minor validity issues that did not fit into Chapter 3 but are regardless important to note for the integrity of the thesis. Stage three of the search process included a validity threat regarding the removal of duplicates. If two full license texts would seem duplicates we would check the two license listing sites’ license pages for further investigation without using an internet archiver. This is a common validity threat on this thesis, that is not relying on an internet archiver on every source possible. Still, archiving more than a thousand license pages and accessing them would have been very slow process in terms of both archiving and accessing.

As can be seen in Chapter 2 the regular expression string was only an exclusion filter. Using an inclusion and exclusion resulted in difficulities to match all of the public software

licenses. In other words it eventually turned out to be faster to match the excludable licenses than the includable licenses. The validity threat lying here is that only using an exclusion filter implicates a majority of the public licenses in our dataset to be public software licenses. An example of difficult to include public software license is the `wtfpl` which includes no evidence of it being a public software license but rather a general public license. However because `wtfpl` is a largely used in software source code as can be seen in Chapter 3. Another examples to back up this choice in exclusion-only are the font licenses that are considered public software licenses. With the exceptions inflating the inclusion regular expression string we eventually decided to only use the aforementioned exclusion filter. Before the decision our inclusion string looked like this:

```
(.*\b(source|software|program|code|module|public(s+)license|ware|
(w+)ware)\b).*
```

As mentioned earlier in the thesis the Wikipedia infobox order of license listing sites plays a heavy role in the literature selection. This manifests as a validity threat for example in removal of duplicates where the duplicates are removed from the lattermost listing site, giving a false sense of the majority of the public licenses coming from the formermost license listing sites like the SPDX. While this might be true due to the high volume of literature from the formermost license listing sites in order of the Wikipedia infobox it is still a threat to validity. Because of this choice in our scope the accuracy of the origins of the licenses in the search stages is not as high as it could be.

FSF license listing site also had some other more minor issues than described earlier. Licenses like `DejaVu` and `DBG-3.0` did have an FSF license page found from the listing site but these pages only offered one single whitespace character as the full license text. Licenses like `CorkForkPL` also contained a whitespace as the full license text but included a note about a software that uses this license. Sometimes the full license text could be found by just clicking the provided hyperlink to the software mentioned which is what we did with `JahiaCSL`. Sometimes it would have required the author to download and unarchive source code to see the full license text or use an internet archiver on top of that due to broken hyperlinks or the software's website being down permanently. We solved this dilemma by deciding to only get the full license text if it was at maximum one click away from the original license listing page. In cases where the license was listed on the FSF license listing page as whitespace the full license text was fetched from the next license listing site in the Wikipedia infobox order if it existed there. For example the full license text for FSF's `MPL` was fetched from GNU under `MPL-1.1`. While we figured out

reproducable rules to our literature selection phase it is fair to note that these are threats to validity regardless of the systematic nature of the remedies presented here.

A more general note on the systematicity of this thesis is due. Systematic does not equal to automatic. The author's human eyesight was for example a major factor to distinguish duplicates in literature selection in search stage three. Licenses were sorted by the Ratcliff and Obershelp, opened all search stage two licenses to tabs on the text editor, switched with keybinds between $n-1$, n and $n+1$ full license texts and removed licenses that the author concluded to be duplicates based on various factors descrbied in Chapter 2. As can be seen the process is systematic and relies heavily on the use of various automated tools but much of the work is also on the responsibility of the author's eyesight, memory and overall judgement which makes this process far from automatic. It is also good to note that the Python script used in Chapter 2 does not work on Windows systems. This was tested to decrease the waiting time of the Ratcliff and Obershelp on a more powerful desktop computer. This is the last and most minor validity threat mentioned in this thesis regarding the literature selection for review.

As mentioned in the beginning of this section, efforts were made to ensure the inclusion of comprehensive set of literature in the search process. However, as with all systematic literature reviews, a comprehensive manual review of all literature would have been a formidable task. Therefore, additional filtering was conducted. This filtering was carried out in two phases, starting with the application of inclusion/exclusion criteria, followed by a second phase focused on evaluating the nature of the public software licenses and conducting a manual review. As a result of this second phase, a set of literature were excluded following a critical appraisal, with documentation and reasoning provided for each section.

As such we can note that the literature selection was done in a sufficient manner.

4.3.2 Limitations in data extraction

The process of data extraction holds great significance in a systematic literature review, as it has a direct impact on the transparency and rationale of the paper. The data extraction approach was shallow due to the data extraction form being relatively small. As mentioned above, not much data could be easily nor verifyiably extracted from our main grey literature, the five license listing sites. Despite the dilligent efforts to eliminate researcher bias, which is a common concern in interpretive methods, it was not feasible

to replicate this work by another individual for cross-referencing purposes. However, the study's validity can still be considered appropriate, due to the transparent steps taken and the use of a short, but well-defined data extraction format.

We still note that because of the lack of common standardized measurements and tooling for them, a considerable amount of personal consideration had to be done to bring the research results of the primary literature into a comparative state.

5 Conclusions

The primary objective of this study was to conduct an extensive review of public licenses in software engineering and offer support to both practitioners and reserachers. The study addressed the following research questions:

RQ1: How many licenses are there in the top five software license listing cites? The results of this study reveal 607 public licenses listed from the SPDX, 38 from the DFSG, 200 from the FSF, 96 from the OSI and 116 from the GNU project with the four latter sites providing most difficulities in fetching and parsing the shortcodes and full license texts with additional problems in overlapping and contradicting information. This resulted in 1057 total licenses.

RQ2: How much is there disagreement in the shortcode names between different public software licenses listing sites? From the 1057 original licenses, some 339 licenses were removed as duplicates using their shortcodes. Reasons for this initial amount of duplicates were discussed and lay background and reasoning for the amount of disagreement in the shortcode names between our license listing sites.

RQ3: How many public licenses in software engineering does there exist? After a manual review of the literature gathered and processed, the study found that there are some 594 unique public software licensed listed that can be obtained systematically.

5.1 Future research

Future research efforts should start at adopting a clear baseline. The definition of free software, open source and everything in between and ultimately the insignificance of the categorization regarding these public software licenses. It is important for the future research to distinguish the terms used yet it is also as important to know that the for example not all free software licenses offer the same freedoms.

In terms of research efforts, a research project that would concentrate on trade-offs and especially the harmful impacts of software projects that do not use a strong copyleft public software license could have significant effect. This project could benefit from the metrics that contain the most used public software licenses and the most significant public software

licensed software, and the effects of using that license with the consideration of the impact of using a different license.

5.2 Future industry efforts

Future industry efforts should focus on trying to learn and understand the practical effects of each public software license used in a software project as a dependency or as an individiual software project licensed under a public software license. Due to the loose definition of open source companies like OpenAI have nothing to do with open source and large language models classes like Meta's Llama 4 are not open source although it is marketed as "Open-Source". Because open source is often seen as altruistic and generally speaking good for the image of a company, the industry continues to see "openwashing" just like greenwashing is used to market something as "green" when it in reality is not. As we have seen in this thesis' results, the definition of open source truly is too loose as it covers even licenses like `Beerware` and `JSON` where payments are in optional alcoholic beverages and restrictions are good and evil. This is why another one of the creators of the original Open Source Definition, Bruce Perens left the initiative (Claburn, 2020). Perens has then proposed a successor to Open Source called Post Open Source. The paradigm combines new public software licenses, existing public software licenses and a practice where one singular organization would gather the money from larger corporations using post open software and distribute it to the contributors of the post open licensed software. It remains to be seen whether or not this will revolutionize the future industry of software engineering.

When it comes to licensing new code from the ground up the near future industry should pay close attention to choosing a sustainable license. As mentioned earlier some of the licenses are not even meant to be court-proof. Licenses like the GPL are constantly trialed by fire in the court and are succesfully defending software freedom even today (Casanova, 2024). Although even more copyleft-oriented public software licenses like the `Watcom` exist, the author recommends licensing software source code under `AGPL-3.0-or-later` where possible due to its strong copyleft attributes and freedoms but mostly the court-proof nature of it as well.

In conclusion, this thesis has provided a systematic review of the current state of research on public licenses in software engineering.Through a systematic literature review, we have identified the amount of unique public software licenses and the methods of obtaining these licenses in a systematic way. The results of this study indicate that the field of

public licenses in software engineering is still immature, with a lack of common standards, definitions, and tooling. However, this also provides an opportunity for further research and development in this area. It is our hope that the findings of this study will serve as a starting point for future research and industry efforts to build the standards and tooling required for more thorough analysis and information sharing, leading to more sustainable software engineering practices.

Bibliography

Balter, B. (2015). *Open source license usage on GitHub.com - The GitHub Blog*. https://web.archive.org/web/20240409120258/https://github.blog/2015-03-09-open-source-license-usage-on-github-com/. Accessed: 2024 April 9.

Bernelin, M. (Aug. 2020). "The compatibility of open/free licences: a legal imbroglio". In: *International Journal of Law and Information Technology* 28.2, pp. 93–111. ISSN: 0967-0769. DOI: 10.1093/ijlit/eaaa010. eprint: https://academic.oup.com/ijlit/article-pdf/28/2/93/33677446/eaaa010.pdf. URL: https://doi.org/10.1093/ijlit/eaaa010.

Casanova, A. (2024). *Failure to Comply with GNU GPL V2 License: Key Takeaways from Orange's Conviction*. https://vaultinum.com/blog/failure-to-comply-with-gnu-gpl-v2-license-key-take. Accessed: 2025 Jul 21.

Claburn, T. (2020). *Bruce Perens quits Open Source Initiative amid row over new data-sharing crypto license: 'We've gone the wrong way with licensing'*. https://www.theregister.com/2020/01/03/osi_cofounder_resigns/. Accessed: 2025 Jul 21.

– (2023). *What comes after open source? Bruce Perens is working on it*. https://www.theregister.com/2023/12/27/bruce_perens_post_open/. Accessed: 2025 Jul 10.

Debian (2024). *The DFSG and Software Licenses*. https://web.archive.org/web/20240415104532/https://wiki.debian.org/DFSGLicenses. Accessed: 2024 April 15.

Duisburg, H. von (2011). "The Enforceability of the General Public License in the US". In: *Zeitschrift der Deutsch-Amerikanischen Juristen-Vereinigung e.V. 36 (2011)* 2, pp. 69–70. URL: https://heinonline.org/HOL/Page?handle=hein.journals/dajvnws2011&id=75.

Ferguson, D. (2006). "Syntar Errors: Why Version 3 of the GNU General Public License Needs Debugging". In: *North Carolina Journal of Law & Technology* 7.2, pp. 397–420. URL: https://heinonline.org/HOL/Page?handle=hein.journals/ncjl7&id=403.

FSF (2024). *Category:License - Free Software Directory*. https://web.archive.org/web/20240409111815/https://directory.fsf.org/wiki/Category:License. Accessed: 2024 April 9.

Garousi, V., Felderer, M., and Mäntylä, M. V. (2019). "Guidelines for including grey literature and conducting multivocal literature reviews in software engineering". In: *Information and Software Technology* 106, pp. 101–121. ISSN: 0950-5849. DOI: https:

//doi.org/10.1016/j.infsof.2018.09.006. URL: https://www.sciencedirect.com/science/article/pii/S0950584918301939.

GitHub (2025). *GitHub Innovation Graph*. https://github.com/github/innovationgraph. Accessed : 2025 July 2nd.

GNU (1996). *What is Free Software?* https://web.archive.org/web/19980126185518/https://www.gnu.org/philosophy/free-sw.html. Accessed: 2024 Feb 8.

– (2023). *Various Licenses and Comments about Them*. https://web.archive.org/web/20240226115940/https://www.gnu.org/licenses/license-list.html. Accessed: 2024 Feb 26.

Hietanen, H. A. (2007). "A License or a Contract, Analyzing the Nature of Creative Commons Licenses". In: *Nordic Intellectual Property Law Review*. Available at SSRN. Posted November 12, 2007; Revised July 20, 2008. URL: https://ssrn.com/abstract=1029366.

Kitchenham, B. and Charters, S. (Jan. 2007). "Guidelines for performing Systematic Literature Reviews in Software Engineering". In: 2.

Kuhn, B. M. (2023). *A Comprehensive Analysis of the GPL Issues With the Red Hat Enterprise Linux RHEL Business Model*. https://web.archive.org/web/20240205080551/https://sfconservancy.org/blog/2023/jun/23/rhel-gpl-analysis/. Accessed: 2024 Feb 5.

Linux Foundation (2024). *SPDX License List*. https://web.archive.org/web/20240412103557/https://spdx.org/licenses/. Accessed: 2024 April 12.

Mahdavi-Hezavehi, S., Galster, M., and Avgeriou, P. (2013). "Variability in quality attributes of service-based software systems: A systematic literature review". In: *Information and Software Technology* 55.2. Special Section: Component-Based Software Engineering (CBSE), 2011, pp. 320–343. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2012.08.010. URL: https://www.sciencedirect.com/science/article/pii/S0950584912001772.

McGrath, M. (2023). *Furthering the evolution of CentOS Stream*. https://web.archive.org/save/https://www.redhat.com/en/blog/furthering-evolution-centos-stream. Accessed: 2024 Feb 5.

Mustonen, M. (2003). "Copyleft—the economics of Linux and other open source software". In: *Information Economics and Policy* 15.1, pp. 99–121. ISSN: 0167-6245. DOI: https://doi.org/10.1016/S0167-6245(02)00090-2. URL: https://www.sciencedirect.com/science/article/pii/S0167624502000902.

Neumayer, E. (1999). *Weak versus Strong Sustainability*. URL: https://ideas.repec.org/b/elg/eebook/1736.html.

Nurmivaara, S. (2023). "Green in software engineering: A systematic literature review". In.

OSI (2008). *GNU General Public License version 3*. https://web.archive.org/web/20240226075409/https://opensource.org/license/gpl-3-0. Accessed: 2024 Feb 26.

– (2024). *Licenses - Open Source Initiative*. https://web.archive.org/web/20240307121412/https://opensource.org/licenses. Accessed: 2024 March 7.

ScanCode (2025). *ScanCode LicenseDB*. https://github.com/aboutcode-org/scancode-licensedb. Accessed : 2025 March 25th.

Stallman, R. (2009). "Viewpoint Why "open source" misses the point of free software". In: *Commun. ACM* 52.6, pp. 31–33. ISSN: 0001-0782. DOI: 10.1145/1516046.1516058. URL: https://doi.org/10.1145/1516046.1516058.

Taguchi, A. (2025). *Taguchi's master's thesis GitHub repository*. https://github.com/akirataguchi115/mscthesis. Accessed: 2025 May 27.

Wikipedians (2024). *MIT License - Wikipedia*. https://web.archive.org/web/20240411081352/https://en.wikipedia.org/wiki/MIT_License. Accessed: 2024 April 11.

– (2025). *Copyleft - Wikipedia*. https://web.archive.org/web/20250616132121/https://en.wikipedia.org/wiki/Copyleft. Accessed: 2025 June 16.

Zhang, H. and Ali Babar, M. (2010). "On searching relevant studies in software engineering". In: *Proceedings of the 14th International Conference on Evaluation and Assessment in Software Engineering*. EASE'10. UK: BCS Learning & Development Ltd., pp. 111–120.

Appendix A Primary literature reviewed, read in full and data extracted

Table A.1: Final list of literature with the inclusion/exclusion criteria applied.

Literature identifier	Shortcode	SPDX	DFSG	FSF	OSI	GNU
L1	0BSD	SPDX			OSI	
L2	996			FSF		
L3	AAL	SPDX			OSI	
L4	Abstyles	SPDX				
L5	ACEL			FSF		
L6	AdaCore-doc	SPDX				
L7	Adobe-2006	SPDX				
L8	Adobe-Display-PostScript	SPDX				
L9	Adobe-Glyph	SPDX				
L10	Adobe-Utopia	SPDX				
L11	ADSL	SPDX				
L12	AFL-1.1	SPDX				
L13	AFL-1.2	SPDX				
L14	AFL-2.0	SPDX				
L15	AFL-2.1	SPDX				
L16	AFL-3.0	SPDX		FSF	OSI	
L17	Afmparse	SPDX				
L18	AGPL-1.0-only	SPDX		FSF		
L19	AGPL-1.0-or-later	SPDX		FSF		
L20	AGPL-3.0-only	SPDX	DFSG	FSF	OSI	
L21	AGPL-3.0-or-later	SPDX		FSF		
L22	Aladdin	SPDX		FSF		GNU
L23	Aladdin-9			FSF		
L24	AMDPLPA	SPDX				
L25	AML	SPDX				
L26	AML-glslang	SPDX				
L27	AMPAS	SPDX				

L28	ANTI-1.3			FSF		
L29	ANTI-1.4			FSF		
L30	ANTLR-PD	SPDX				
L31	ANTLR-PD-fallback	SPDX				
L32	Apache-1.0	SPDX		FSF		
L33	Apache-1.1	SPDX		FSF	OSI	
L34	Apache-2.0	SPDX	DFSG	FSF	OSI	
L35	APAFML	SPDX				
L36	APL-1.0	SPDX			OSI	
L37	App-s2p	SPDX				
L38	APSL-1.0	SPDX		FSF		
L39	APSL-1.1	SPDX		FSF		
L40	APSL-1.2	SPDX		FSF		
L41	APSL-2.0	SPDX	DFSG	FSF	OSI	
L42	Arphic-1999	SPDX				
L43	Artistic-1.0	SPDX		FSF	OSI	
L44	Artistic-1.0-cl8	SPDX				
L45	Artistic-1.0-Perl	SPDX		FSF	OSI	
L46	Artistic-2.0	SPDX	DFSG	FSF	OSI	
L47	ASWF-Digital-Assets-1.0	SPDX				
L48	ASWF-Digital-Assets-1.1	SPDX				
L49	ATTPublicLicense					GNU
L50	Baekmuk	SPDX				
L51	Bahyph	SPDX				
L52	Barr	SPDX				
L53	bcrypt-Solar-Designer	SPDX				
L54	Beerware	SPDX				
L55	BerkeleyDB					GNU
L56	Bitstream-Charter	SPDX				
L57	Bitstream-Vera	SPDX				
L58	BitTorrent-1.0	SPDX				
L59	BitTorrent-1.1	SPDX		FSF		
L60	blessing	SPDX				
L61	BlueOak-1.0.0	SPDX			OSI	
L62	Boehm-GC	SPDX				

L63	Borceux	SPDX			
L64	Brian-Gladman-2-Clause	SPDX			
L65	Brian-Gladman-3-Clause	SPDX			
L66	BSD-1-Clause	SPDX		FSF	OSI
L67	BSD-2-Clause	SPDX		FSF	
L68	BSD-2-Clause-Darwin	SPDX			
L69	BSD-2-Clause-FreeBSD			FSF	
L70	BSD-2-Clause-Patent	SPDX			OSI
L71	BSD-2-Clause-Views	SPDX			
L72	BSD-3-Clause	SPDX	DFSG	FSF	OSI
L73	BSD-3-Clause-acpica	SPDX			
L74	BSD-3-Clause-Attribution	SPDX			
L75	BSD-3-Clause-Clear	SPDX		FSF	
L76	BSD-3-Clause-flex	SPDX			
L77	BSD-3-Clause-HP	SPDX			
L78	BSD-3-Clause-LBNL	SPDX			OSI
L79	BSD-3-Clause-Modification	SPDX			
L80	BSD-3-Clause-No-Military-License	SPDX			
L81	BSD-3-Clause-No-Nuclear-License	SPDX			
L82	BSD-3-Clause-No-Nuclear-License-2014	SPDX			
L83	BSD-3-Clause-No-Nuclear-Warranty	SPDX			
L84	BSD-3-Clause-Open-MPI	SPDX			
L85	BSD-3-Clause-Sun	SPDX			
L86	BSD-4-Clause	SPDX		FSF	
L87	BSD-4-Clause-Shortened	SPDX			
L88	BSD-4-Clause-UC	SPDX			
L89	BSD-4.3RENO	SPDX			
L90	BSD-4.3TAHOE	SPDX			
L91	BSD-Advertising-Acknowledgement	SPDX			
L92	BSD-Attribution-HPND-disclaimer	SPDX			
L93	BSD-Inferno-Nettverk	SPDX			
L94	BSD-Protection	SPDX			
L95	BSD-Source-beginning-file	SPDX			
L96	BSD-Source-Code	SPDX			

L97	BSD-Systemics	SPDX			
L98	BSD-Systemics-W3Works	SPDX			
L99	BSL-1.0	SPDX		FSF	OSI
L100	BUSL-1.1	SPDX			
L101	bzip2-1.0.6	SPDX			
L102	C-UDA-1.0	SPDX			
L103	CAL-1.0	SPDX			OSI
L104	CAL-1.0-Combined-Work-Exception	SPDX			
L105	Caldera	SPDX			
L106	Caldera-no-preamble	SPDX			
L107	CATOSL-1.1	SPDX			
L108	CDDL-1.0	SPDX		FSF	
L109	CDDL-1.1	SPDX			
L110	CDLA-Permissive-1.0	SPDX			
L111	CDLA-Permissive-2.0	SPDX			
L112	CDLA-Sharing-1.0	SPDX			
L113	CECILL-1.0	SPDX			
L114	CECILL-1.1	SPDX			
L115	CECILL-2.0	SPDX		FSF	
L116	CECILL-2.1	SPDX			OSI
L117	CECILL-B	SPDX			
L118	Cecill-B-v1			FSF	
L119	CECILL-C	SPDX			
L120	Cecill-C-v1			FSF	
L121	CERN-OHL-1.1	SPDX			
L122	CERN-OHL-1.2	SPDX			
L123	CERN-OHL-P-2.0	SPDX			OSI
L124	CERN-OHL-S-2.0	SPDX			OSI
L125	CERN-OHL-W-2.0	SPDX			OSI
L126	CFITSIO	SPDX			
L127	check-cvs	SPDX			
L128	checkmk	SPDX			
L129	ClArtistic	SPDX		FSF	
L130	Clips	SPDX			
L131	CMU-Mach	SPDX			

L132	CMU-Mach-nodoc	SPDX			
L133	CNRI			FSF	
L134	CNRI-Jython	SPDX			
L135	CNRI-Python	SPDX			OSI
L136	CNRI-Python-GPL-Compatible	SPDX			
L137	COIL-1.0	SPDX			
L138	Commons-Clause			FSF	
L139	Condor-1.1	SPDX		FSF	
L140	copyleft-next-0.3.0	SPDX			
L141	copyleft-next-0.3.1	SPDX			
L142	Cornell-Lossless-JPEG	SPDX			
L143	CPAL-1.0	SPDX	DFSG	FSF	OSI
L144	CPL-1.0	SPDX	DFSG	FSF	OSI
L145	CPOL-1.02	SPDX		FSF	
L146	Cronyx	SPDX			
L147	Crossword	SPDX			
L148	CryptixGL			FSF	
L149	CrystalStacker	SPDX			
L150	CUA-OPL-1.0	SPDX			
L151	Cube	SPDX			
L152	curl	SPDX		FSF	
L153	cvw				OSI
L154	D-FSL-1.0	SPDX			
L155	DEC-3-Clause	SPDX			
L156	Design-Science-L			FSF	
L157	diffmark	SPDX			
L158	DL-DE-BY-2.0	SPDX			
L159	DL-DE-ZERO-2.0	SPDX			
L160	DOC	SPDX			
L161	Dotseqn	SPDX			
L162	DRL-1.0	SPDX			
L163	DRL-1.1	SPDX			
L164	DSDP	SPDX			
L165	dtoa	SPDX			
L166	dvipdfm	SPDX			

L167	ECL-1.0	SPDX				
L168	ECL-2.0	SPDX		FSF	OSI	
L169	ECos-2.0			FSF	OSI	
L170	EFL-1.0	SPDX			OSI	
L171	EFL-2.0	SPDX		FSF	OSI	
L172	eGenix	SPDX				
L173	Elastic-2.0	SPDX				
L174	Entessa	SPDX				
L175	EPICS	SPDX		FSF		
L176	EPL-1.0	SPDX	DFSG	FSF	OSI	
L177	EPL-2.0	SPDX		FSF	OSI	
L178	ErlPL-1.1	SPDX		FSF		
L179	EUDatagrid	SPDX		FSF	OSI	GNU
L180	EUPL-1.0	SPDX				
L181	EUPL-1.1	SPDX		FSF	OSI	
L182	EUPL-1.2	SPDX			OSI	
L183	Eurosym	SPDX				
L184	Expat			FSF		GNU
L185	Fair	SPDX				
L186	FAL		DFSG			
L187	FBM	SPDX				
L188	FDK-AAC	SPDX				
L189	Ferguson-Twofish	SPDX				
L190	Frameworx-1.0	SPDX				
L191	FreeBSD-DOC	SPDX				
L192	FreeImage	SPDX				
L193	FSFAP	SPDX		FSF		
L194	FSFAP-no-warranty-disclaimer	SPDX				
L195	FSFUL	SPDX				
L196	FSFULLR	SPDX				
L197	FSFULLRWD	SPDX				
L198	FTL	SPDX		FSF		
L199	Furuseth	SPDX				
L200	fwlw	SPDX				
L201	GCR-docs	SPDX				

L202	GD	SPDX				
L203	Giftware	SPDX				
L204	GL2PS	SPDX		FSF		
L205	Glide	SPDX				
L206	Glulxe	SPDX				
L207	GLWTPL	SPDX				
L208	gnuplot	SPDX		FSF		GNU
L209	GPL-1.0-only	SPDX		FSF		
L210	GPL-1.0-or-later	SPDX		FSF		
L211	GPL-2.0-only	SPDX		FSF		
L212	GPL-2.0-or-later	SPDX		FSF		
L213	GPL-3.0-only	SPDX	DFSG	FSF	OSI	
L214	GPL-3.0-or-later	SPDX		FSF		
L215	GPL-PA			FSF		
L216	Graphics-Gems	SPDX				
L217	gSOAP-1.3b	SPDX				
L218	gtkbook	SPDX				
L219	HaskellReport	SPDX				
L220	hdparm	SPDX				
L221	HESSLA			FSF		GNU
L222	Hippocratic-2.1	SPDX				
L223	HP-1986	SPDX				
L224	HP-1989	SPDX				
L225	HPND	SPDX		FSF		GNU
L226	HPND-DEC	SPDX				
L227	HPND-doc	SPDX				
L228	HPND-doc-sell	SPDX				
L229	HPND-export-US	SPDX				
L230	HPND-export-US-modify	SPDX				
L231	HPND-Fenneberg-Livingston	SPDX				
L232	HPND-INRIA-IMAG	SPDX				
L233	HPND-Kevlin-Henney	SPDX				
L234	HPND-Markus-Kuhn	SPDX				
L235	HPND-MIT-disclaimer	SPDX				
L236	HPND-Pbmplus	SPDX				

L237	HPND-sell-MIT-disclaimer-xserver	SPDX				
L238	HPND-sell-regexpr	SPDX				
L239	HPND-sell-variant	SPDX				
L240	HPND-sell-variant-MIT-disclaimer	SPDX				
L241	HPND-UC	SPDX				
L242	HTMLTIDY	SPDX				
L243	IBM-pibs	SPDX				
L244	IBMPL					GNU
L245	ICU	SPDX			OSI	
L246	IEC-Code-Components-EULA	SPDX				
L247	IJG	SPDX		FSF		GNU
L248	IJG-short	SPDX				
L249	ImageMagick	SPDX				
L250	iMatix	SPDX		FSF		GNU
L251	imlib					GNU
L252	Imlib2	SPDX		FSF		
L253	Info-ZIP	SPDX		FSF		
L254	informal					GNU
L255	Inner-Net-2.0	SPDX				
L256	Intel	SPDX		FSF		GNU
L257	Intel-ACPI	SPDX		FSF		
L258	Interbase-1.0	SPDX				
L259	IPA	SPDX		FSF	OSI	
L260	IPL-1.0	SPDX	DFSG	FSF		
L261	ISC	SPDX	DFSG	FSF	OSI	GNU
L262	ISC-Veillard	SPDX				
L263	JahiaCSL			FSF		
L264	Jam	SPDX			OSI	
L265	JasPer-2.0	SPDX				
L266	JOSL-1.0			FSF		
L267	JPL-image	SPDX				
L268	JPNIC	SPDX				
L269	JSON	SPDX	DFSG	FSF		GNU
L270	Kastrup	SPDX				
L271	Kazlib	SPDX				

L272	Knuth-CTAN	SPDX				
L273	LAL-1.2	SPDX				
L274	LAL-1.3	SPDX		FSF		
L275	LaTeX ecfonts			FSF		
L276	Latex2e	SPDX				
L277	Latex2e-translated-notice	SPDX				
L278	Leptonica	SPDX				
L279	LGPL-2.0-only	SPDX		FSF	OSI	
L280	LGPL-2.0-or-later	SPDX		FSF		
L281	LGPL-2.1-only	SPDX		FSF		
L282	LGPL-2.1-or-later	SPDX		FSF		
L283	LGPL-3.0-only	SPDX	DFSG	FSF	OSI	
L284	LGPL-3.0-or-later	SPDX		FSF		
L285	LGPLLR	SPDX		FSF		
L286	Lha			FSF		GNU
L287	Libpng	SPDX				
L288	libpng-2.0	SPDX				
L289	libselinux-1.0	SPDX				
L290	libtiff	SPDX				
L291	libutil-David-Nugent	SPDX				
L292	LiLiQ-P-1.1	SPDX			OSI	
L293	LiLiQ-R-1.1	SPDX			OSI	
L294	LiLiQ-Rplus-1.1	SPDX			OSI	
L295	Linux-man-pages-1-para	SPDX				
L296	Linux-man-pages-copyleft	SPDX				
L297	Linux-man-pages-copyleft-2-para	SPDX				
L298	Linux-man-pages-copyleft-var	SPDX				
L299	Linux-OpenIB	SPDX				
L300	LLGPL			FSF		
L301	LOOP	SPDX				
L302	LPD-document	SPDX				
L303	LPL-1.0	SPDX				
L304	LPL-1.02	SPDX		FSF		
L305	LPPL-1.0	SPDX				
L306	LPPL-1.1	SPDX				

L307	LPPL-1.2	SPDX		FSF		
L308	LPPL-1.3a	SPDX		FSF		
L309	LPPL-1.3c	SPDX		FSF	OSI	
L310	lsof	SPDX				
L311	Lua license			FSF		
L312	Lucida-Bitmap-Fonts	SPDX				
L313	LZMA-SDK-9.11-to-9.20	SPDX				
L314	LZMA-SDK-9.22	SPDX				
L315	Mackerras-3-Clause	SPDX				
L316	Mackerras-3-Clause-acknowledgment	SPDX				
L317	magaz	SPDX				
L318	mailprio	SPDX				
L319	MakeIndex	SPDX				
L320	Martin-Birgmeier	SPDX				
L321	McPhee-slideshow	SPDX				
L322	metamail	SPDX				
L323	Minpack	SPDX				
L324	MirOS	SPDX	DFSG	FSF	OSI	
L325	MIT	SPDX	DFSG		OSI	
L326	MIT-0	SPDX			OSI	
L327	MIT-advertising	SPDX				
L328	MIT-CMU	SPDX				
L329	MIT-enna	SPDX				
L330	MIT-feh	SPDX				
L331	MIT-Festival	SPDX				
L332	MIT-Modern-Variant	SPDX				
L333	MIT-open-group	SPDX				
L334	MIT-testregex	SPDX				
L335	MIT-Wu	SPDX				
L336	MITNFA	SPDX				
L337	MMIXware	SPDX				
L338	Modified X11			FSF		
L339	ModifiedBSD					GNU
L340	Motosoto	SPDX			OSI	
L341	MPEG-SSG	SPDX				

L342	mpi-permissive	SPDX				
L343	mpich2	SPDX				
L344	MPL			FSF		GNU
L345	MPL-1.0	SPDX			OSI	
L346	MPL-1.1	SPDX		FSF	OSI	
L347	MPL-2.0	SPDX	DFSG	FSF	OSI	
L348	MPL-2.0-no-copyleft-exception	SPDX				
L349	mplus	SPDX				
L350	MS-LPL	SPDX				
L351	MS-PL	SPDX		FSF	OSI	
L352	MS-RL	SPDX		FSF	OSI	
L353	Ms-SS			FSF		
L354	MTLL	SPDX				
L355	MulanPSL-1.0	SPDX				
L356	MulanPSL-2.0	SPDX			OSI	
L357	Multics	SPDX			OSI	
L358	Mup	SPDX				
L359	NAIST-2003	SPDX				
L360	NASA-1.3	SPDX		FSF		
L361	Naumen	SPDX				
L362	NBPL-1.0	SPDX				
L363	NCGL-UK-2.0	SPDX				
L364	NCSA	SPDX		FSF	OSI	GNU
L365	Net-SNMP	SPDX				
L366	NetCDF	SPDX				
L367	NetscapeJavaScript					GNU
L368	Newsletr	SPDX				
L369	NGPL	SPDX		FSF	OSI	
L370	NICTA-1.0	SPDX				
L371	NIST-PD	SPDX				
L372	NIST-PD-fallback	SPDX				
L373	NIST-Software	SPDX				
L374	NLPL	SPDX				
L375	Nokia	SPDX		FSF	OSI	GNU
L376	NoLicense					GNU

L377	NOSL	SPDX		FSF		GNU
L378	Noweb	SPDX				
L379	NPL-1.0	SPDX				
L380	NPL-1.1	SPDX		FSF		
L381	NPOSL-3.0	SPDX			OSI	
L382	NRL	SPDX				
L383	NTP	SPDX			OSI	
L384	NTP-0	SPDX				
L385	O-UDA-1.0	SPDX				
L386	OCCT-PL	SPDX				
L387	OCL-1.0			FSF		
L388	OCLC-2.0	SPDX				
L389	Oculus VR Rift SDK License			FSF		
L390	ODbL-1.0	SPDX		FSF		
L391	OFFIS	SPDX				
L392	OFL-1.0	SPDX				
L393	OFL-1.0-no-RFN	SPDX				
L394	OFL-1.0-RFN	SPDX				
L395	OFL-1.1	SPDX	DFSG	FSF	OSI	
L396	OFL-1.1-no-RFN	SPDX				
L397	OFL-1.1-RFN	SPDX				
L398	OGC-1.0	SPDX				
L399	OGL-Canada-2.0	SPDX				
L400	OGTSL	SPDX			OSI	
L401	OLDAP-1.1	SPDX				
L402	OLDAP-1.2	SPDX				
L403	OLDAP-1.3	SPDX				
L404	OLDAP-1.4	SPDX				
L405	OLDAP-2.0	SPDX				
L406	OLDAP-2.0.1	SPDX				
L407	OLDAP-2.1	SPDX				
L408	OLDAP-2.2	SPDX				
L409	OLDAP-2.2.1	SPDX				
L410	OLDAP-2.2.2	SPDX				
L411	OLDAP-2.3	SPDX		FSF		

L412	OLDAP-2.4	SPDX				
L413	OLDAP-2.5	SPDX				
L414	OLDAP-2.6	SPDX				
L415	OLDAP-2.7	SPDX		FSF		
L416	OLDAP-2.8	SPDX		FSF	OSI	
L417	oldOpenLDAP					GNU
L418	OLFL-1.3	SPDX			OSI	
L419	OML	SPDX				
L420	Open Publication License v1.0			FSF		
L421	OpenPBS-2.3	SPDX	DFSG			
L422	OpenSSL	SPDX		FSF		GNU
L423	OpenSSL-standalone	SPDX				
L424	OpenVision	SPDX				
L425	OPL-1.0	SPDX	DFSG	FSF		
L426	OPL-UK-3.0	SPDX				
L427	OPUBL-1.0	SPDX				
L428	OriginalBSD					GNU
L429	OSET-PL-2.1	SPDX			OSI	
L430	OSL					GNU
L431	OSL-1.0	SPDX			OSI	
L432	OSL-1.1	SPDX	DFSG			
L433	OSL-2.0	SPDX				
L434	OSL-2.1	SPDX			OSI	
L435	OSL-3.0	SPDX		FSF	OSI	
L436	PADL	SPDX				
L437	Parity-6.0.0	SPDX				
L438	Parity-7.0.0	SPDX				
L439	PerlLicense					GNU
L440	Phorum-2.0			FSF		
L441	PHP-3.0	SPDX		FSF	OSI	
L442	PHP-3.01	SPDX		FSF	OSI	
L443	PINE			FSF		GNU
L444	Pixar	SPDX				
L445	Plexus	SPDX				
L446	pnmstitch	SPDX				

L447	PolyForm-Noncommercial-1.0.0	SPDX				
L448	PolyForm-Small-Business-1.0.0	SPDX				
L449	PostgreSQL	SPDX			OSI	
L450	PPL3a					GNU
L451	PSF-2.0	SPDX			OSI	
L452	psfrag	SPDX				
L453	psutils	SPDX				
L454	PublicDomain			FSF		GNU
L455	Python-1.6a2			FSF		
L456	Python-2.0	SPDX				
L457	Python-2.0.1	SPDX		FSF		
L458	python-ldap	SPDX				
L459	Qhull	SPDX				
L460	QPL-1.0	SPDX	DFSG	FSF	OSI	
L461	QPL-1.0-INRIA-2004	SPDX				
L462	radvd	SPDX				
L463	Rdisc	SPDX				
L464	RHeCos-1.1	SPDX		FSF		
L465	RPL-1.1	SPDX			OSI	
L466	RPL-1.3			FSF		
L467	RPL-1.5	SPDX			OSI	
L468	RPSL-1.0	SPDX	DFSG	FSF	OSI	
L469	RSA-MD	SPDX				
L470	RSCPL	SPDX			OSI	
L471	Ruby	SPDX		FSF		GNU
L472	SAX-PD	SPDX				
L473	SAX-PD-2.0	SPDX				
L474	Saxpath	SPDX				
L475	SCEA	SPDX				
L476	SchemeReport	SPDX				
L477	Scilab-old			FSF		
L478	Scratch			FSF		GNU
L479	SCSL-2.8			FSF		
L480	Sendmail	SPDX		FSF		
L481	Sendmail-8.23	SPDX				

L482	SGI-B-1.0	SPDX				
L483	SGI-B-1.1	SPDX				
L484	SGI-B-2.0	SPDX		FSF		
L485	SGI-OpenGL	SPDX				
L486	SGIFreeB					GNU
L487	SGP4	SPDX				
L488	SHL-0.5	SPDX				
L489	SHL-0.51	SPDX				
L490	SimPL-2.0	SPDX			OSI	
L491	SimpleM			FSF		
L492	SimplePermissive			FSF		
L493	SimplePermissiveNoNonWarranty			FSF		
L494	SISSL	SPDX		FSF	OSI	GNU
L495	SISSL-1.2	SPDX		FSF		
L496	SL	SPDX				
L497	Sleepycat	SPDX		FSF	OSI	
L498	SMLNJ	SPDX		FSF		
L499	SMPPL	SPDX				
L500	SNIA	SPDX				
L501	snprintf	SPDX				
L502	softSurfer	SPDX				
L503	Soundex	SPDX				
L504	Spencer-86	SPDX		FSF		
L505	Spencer-94	SPDX				
L506	Spencer-99	SPDX				
L507	spin		DFSG			
L508	SPL-1.0	SPDX		FSF	OSI	
L509	Squeak-old			FSF		
L510	ssh-keyscan	SPDX				
L511	SSH-OpenSSH	SPDX				
L512	SSH-short	SPDX				
L513	SSLeay-standalone	SPDX				
L514	SSPL-1.0	SPDX				
L515	SSSCFR-1.1			FSF		
L516	StandardMLofNJ					GNU

L517	SugarCRM-1.1.3	SPDX				
L518	Sun-PPP	SPDX				
L519	SunPro	SPDX				
L520	SWL	SPDX				
L521	swrule	SPDX				
L522	Symlinks	SPDX				
L523	TAPR-OHL-1.0	SPDX				
L524	TCL	SPDX		FSF		
L525	TCP-wrappers	SPDX				
L526	TermReadKey	SPDX				
L527	TGPPL-1.0	SPDX		FSF		
L528	THL-1.1			FSF		
L529	TMate	SPDX				
L530	TORQUE-1.1	SPDX				
L531	TOSL	SPDX				
L532	TPDL	SPDX				
L533	TPL-1.0	SPDX				
L534	TrueCrypt			FSF		
L535	TTWL	SPDX				
L536	TTYP0	SPDX				
L537	TU-Berlin-1.0	SPDX				
L538	TU-Berlin-2.0	SPDX				
L539	UCAR	SPDX				
L540	UCL-1.0	SPDX			OSI	
L541	ulem	SPDX				
L542	UMich-Merit	SPDX				
L543	Unicode-3.0	SPDX				
L544	Unicode-DFS-2012			FSF		
L545	Unicode-DFS-2015	SPDX			OSI	
L546	Unicode-DFS-2016	SPDX				
L547	Unicode-TOU	SPDX				
L548	UnixCrypt	SPDX				
L549	Unlicense	SPDX			OSI	GNU
L550	UPL-1.0	SPDX			OSI	
L551	URT-RLE	SPDX				

L552	UtahPublicLicense					GNU
L553	Vim	SPDX				GNU
L554	VOSTROM	SPDX				
L555	VSL-0.1				OSI	
L556	VSL-1.0	SPDX				
L557	W3C	SPDX				GNU
L558	W3C-19980720	SPDX				
L559	W3C-20150513	SPDX			OSI	
L560	w3m	SPDX				
L561	Watcom-1.0	SPDX				
L562	WebM					GNU
L563	Widget-Workshop	SPDX				
L564	Wsuipa	SPDX				
L565	WTFPL	SPDX	DFSG			GNU
L566	wxWindows				OSI	
L567	x-oz		DFSG			
L568	X11	SPDX				
L569	X11-distribute-modifications-variant	SPDX				
L570	X11License					GNU
L571	Xdebug-1.03	SPDX				
L572	Xerox	SPDX				
L573	Xfig	SPDX				
L574	XFree86-1.1	SPDX				
L575	xinetd	SPDX				GNU
L576	xkeyboard-config-Zinoviev	SPDX				
L577	xlock	SPDX				
L578	Xnet	SPDX			OSI	
L579	xpp	SPDX				
L580	XSkat	SPDX				
L581	Yahoo					GNU
L582	YPL-1.0	SPDX				
L583	YPL-1.1	SPDX				
L584	Zed	SPDX				
L585	Zeeff	SPDX				
L586	Zend-2.0	SPDX				

L587	Zimbra-1.3	SPDX				
L588	Zimbra-1.4	SPDX				
L589	Zlib	SPDX	DFSG		OSI	GNU
L590	zlib-acknowledgement	SPDX				
L591	Zope					GNU
L592	ZPL-1.1	SPDX				
L593	ZPL-2.0	SPDX				
L594	ZPL-2.1	SPDX				