Master's thesis

Master's Programme in Computer Science

Public licenses in Software Engineering: A Multivocal Literature Review

Akira Taguchi

June 5, 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 Faculty of Science	Koulutusohjelma — Utbildningsprogram — Study programme 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 Master's thesis	Aika — Datum — Month and year June 5, 2025	Sivumäärä — Sidoantal — Number of pages 28 pages, 18 appendix pages

Tiivistelmä — Referat — Abstract

Context: Public licenses are central to the distribution of works in software engineering. For example in open source there must be an appropriate PCL attached to the source code in order for open-source software to be freely available for possible modification and redistribution. Understanding PCLs can be difficult. This could stem from the legal nature of the license texts and the large number of already-existing PCLs. As a result some actions made within the boundaries of the PCLs may come as a surprise to the public.

Objective: The primary goal of this research is to conduct a multivocal literature review of the current state of PCLs in software engineering, the evaluation of the them and the evidence level of the research. The research aims to provide a novel perspective on relevant licenses and to extract key findings through a rigorous literature review process. This study has two main viewpoints: to provide rigorous research on PCLs to the academic field and to provide insights to the professional field of software engineering on PCLs. The grand goal of this thesis is to raise awareness of the importance of PCLs so that more licensers would make the correct choices based on their situations and needs in a mindful way.

Method: The search strategy examined 6666 sources, found through websites that list PCLs and ad-hoc searches. Applying inclusion and exclusion criteria resulted in the selection of 666 sources, which made relevant contributions related to PCLs in software engineering.

Results:

Conclusions:

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

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.

much love to artemis, sami nurmivaara, prof männistö and prof mäntylä

thanks to def for borrowing gpt4. thanks to rashid and barunes for sending me software licensing related videos and news. thanks to iikka and joonas for giving a hint for using a license database. thanks to suvi for supporting my work with the thesis from the beginning and enduring my mental health issues.

Contents

1 Introduction

use the word public license instead of PCL, make fixes per perens' other comments. Free software license != strong copyleft software license

PCLs play a central role to the distribution of works in software engineering. For example in open source there must be an appropriate PCL 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 must also be considered important in the same context.

Public license is defined by Wikipedia with the following words (Wikipedians, 2024a):

> ”A public license is a copyright license where the licensees are not limited. Examples include free content, open content, Creative Commons, free software and open source licences.”

Understanding PCLs can be difficult. This could stem from the legal nature of the license texts and the large number of already-existing PCLs. 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 PCLs leaves too much room for interpretation. In June 21, 2023 International Business Machines' (IBM) Red Hat seemingly violated a PCL, the GNU General Public License version 2 (GPL-2.0) (Kuhn, 2023) (McGrath, 2023). This was an unpleasant surprise to the public since 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.

Regardless, IBM's Red Hat essentially rendered the previously public Red Hat Enterprise Linux (RHEL) into proprietary software. If the licenses would be more easily understood the proprietarization of RHEL would have been less of a surprise to the users.

On top of PCL details, software engineers in general have a tough time understanding the basic goals of PCLs used in software engineering. In the instance of the RHEL incident it would not have been a big surprise to software engineers if they would have known about other licenses and what they try to achieve or how old is GPLv2 and why it has been succeeded by GNU General Public License version 3 (GPL-3.0).

This thesis' goal is to contribute into the solving these problems in a structured manner. First 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 why the conversely the definitions are the most stabile ones in this area. Second we take a deep dive into the PCLs through a multivocal literature review. To make more information available, a mapping study connected to the terminology scope defined in the first step is needed. Third includes our own suggestions and basic knowledge for professionals and academics in the industry to enhance the understanding of PCLs 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.

1.1 Research goal, questions and contributions

The primary goal of this research is to conduct a multivocal literature review of the current state of PCLs in software engineering, the evaluation of the them and the evidence level of the research. The research aims to provide a novel perspective on relevant licenses and to extract key findings through a rigorous literature review process. The research questions of the review are:

- RQ1: How many PCLs in software engineering does there exist?
- RQ3: What is the average length of a PCL in software engineering?
- RQ3: What are the most common components seen in PCLs in software engineering?
- RQ4: What are the most common changes made to PCLS in software engineering?

Terms such as open source, source code, software freedom and other vocabulary must be defined in the scope of this thesis. Section 1.3 will examine this plethora of of terminology and definitions and will be used to establish a sound basis for discussing this broad subject.

This study has two main viewpoints. The first one is to provide rigorous multivocal research on PCLs to the academic field. Because this thesis already does the multivocal work on PCLs 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 PCLs. Hopefully this makes conversation on PCLs 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 PCLs in software engineering.

1.2 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.

1.3 Background and terminology of PCLs

The current terminology is used with different definitions which leads to inconsistencies 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). This is because their definitions on open source and free software differ from each other. Some parts of the two definitions are even 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 PCLs in software engineering, which distinguishes our investigation from the broader topic of PCLs or the copyright law. This includs also PCLs that are not approved by the FSF nor OSI hence not falling under the group of FLOSS licenses. The term ”copyleft” is defined by Mustonen, 2003 in the following way:

> ”Copyeft 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.”

This is why the term is often used in the context of free software.

In this section we aim to increase the accessibility of our discussion by providing a concise overview of the background of the field of PCLs and the terms we employ.

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

These domains include:

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

The primary aim of this study is to investigate PCLs in software engineering process. However, it is important to acknowledge that PCLs in software engineering are only aspect of PCLs. These additional dimensions are crucial in adoption and implementation of PCLs 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 PCLs 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 PCLs are generally meant to be used as valid legal instruments. The question whether or not a PCL 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 PCLs has seen discussion in the academic field of law since the dawn of PCLs 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 PCLs in software engineering in public are either open-source licenses or free-software licenses and since both paradigms are driven by different organizations with very different goals and values, it is understandable how non-standardized the terminology in the scope of PCL in SE 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.

To provide an understanding of the terminology used in this thesis, a Venn diagram is presented in Figure 1.1, which contextualizes the non-standardized terminology within the PCL scope as a whole. This perspective provides an increased understanding of where different subdomains fall in the larger picture of PCLs. Furthermore it is essential to note that PCLs in software engineering encompasses different aspects that require a closer examination.

Let us explore further the differences and similarities between open source and free software at the software engineering level of PCLs. This is a crucial step since we can see from the approximation in Figure 1.1 that the majority of PLCs are either free software, open source 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

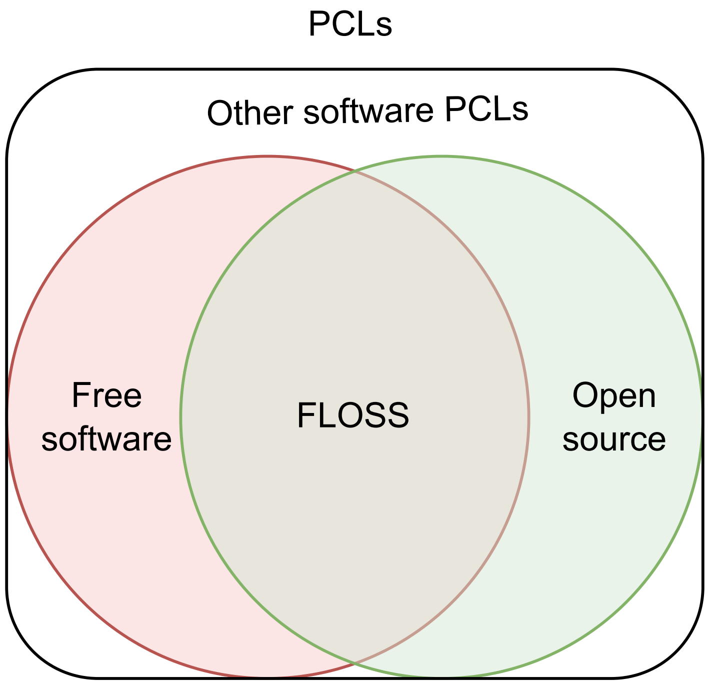

Figure 1.1: PCLs in software engineering

and what does not. For example a new software PCL will not classify as free software nor open source until the corresponding organization has acknowledged the software PCL as either free software, open source or neither. If a PCL is accepted by both FSF and OSI it will fall under the term FLOSS. If a PCL gets accepted by neither of the organization or it gets rejected by both organizations it will fall under other software PCLs in the Figure 1.1. In general 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 does not 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 PCLs 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 PCLs regarding documentation, media and all other non-software targeted PCLs.

The quest to categorize every software PCL 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.

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.

stage 1 licenses amount before rough deduplication could be a nice metric to say for my excel efforts. duplicates.txt can be just a one-time pass of the difflib duplicate thingy. ok now ill go through once the excluded and included licenses one by one and note any broader scale "mistakes" in the search string. gotta add sources on top of source

at the end of this section i will have to document the timestamp at which point i accessed the docs of scancode-licesendb since. actually i might be a good idea to create a wayback machine attempt to capturing the state of the licensedb rn. nvm great wayback machine is not viable solution for that. just stating here that i downloaded the licensedb on 2025 mar 25 15:30

it's good to note that when bumping into the missing license of attpubliclicense which was from gnu, it turned out that the license listing site doesnt state the license content whatsoever. i had to just put the comment of the license into manual licenses.

manual/missing licenses: check which shortcodes of missing licenses does stage2-3.py output. check from sheets where does the shrotcode originate from. ctrl + f that listing site. copy the full license to manual-licenses/shortcode.txt and re-run the stage2-3.py until no longer missing licenses appear in console.

remember to count here how many missing licenses / manual licenses had to be done manually

could be also a good idea to note here how many missing licenses came from which site. gnu and fsf seem to be the winners here.

cvw was not found with the python script. i checked from the excel that its from the osi but osi didnt have it anymore by the time i got there. at some point between fetching the data and the re-fetching the license content from the original site MITRE had voluntarily retired the cvw license. it was found from the scancode licensedb with the name cvwl instead. dejavu and dbg-3.0 were also two other licenses that contained a space. this might indicate that the space is an accident that its simply just not found from a license listing site x. its also good to note that the python script was decided to be an valid approach since many of the licenses were actually found with the shortcode from the licensedb scancode. fetching 700 licenses by hand would have had time and validity issues. wayback machine could have been used to do the actual searching as well. this is unfortunately a validity issue but at least the source is available in wayback machine.

let it be said, that when looking for manual licenses, if there existed two sites which listed the same license, the leftmost was chosen, as per the order of the wikipedia mit article

it seems like i have manually added licenses and invented shortcodes for them AT LEAST in DFSG license listing. FAL is a good example.

For example licenses like CorkForkPL from FSF are just empty licenses. CorkForkPL is used however in MighTyD project but the license would have to be seen from a downloaded project or something like that. A new scope: only licenses that are one (1) click away from the initial license landing page can be copypasted to the manual licenses. just like the JahiaCSL has a URL on FSF for the license new location although the FSF page is empty.

licenses like MPL exist on FSF and GNU. it was not easily found from FSF (empty with links to programs using this) so it was gotten from GNU which was labeled as MPL1.1, which the FSF DOES have so i just boldly went with that. threat to validity.

it could be a good idea to mention how many missing licenses came out of which sites. or it could be out of scope. i can just make a validity threat and say with face value that most of them were from FSF, GNU in that order. Python license seems just straight up an accident on FSF's side. scope is not to fix the documentation problems of the 5 organizations though so I'll leave it just like that and mention the possibility of it being just an accident.

so. now i figured ill take another validity threat L and decide to remove duplicates by human eyesight and choose the one left over from two seeming duplicates by pure human eyesight. i will document the ones that were chosen over the other duplicate and call it a day. for now. i gotta write number + shortcode to find duplicates and document them.

it would be good to note in conclusions that while im stripping away here the errorish licenses and "duplicates". the ones used mostly are different from those that are actually legally valid historically or even those that are not meant to be legally valid (MIT). whilst im doing this systematically it doesnt mean that all parts of it can be done automatically or by automation. human touch in for example checking the incl excl criteria and removing duplicates by combining and creating the weight value of shortcode + license name + license text is done basically by a human with the help of python first sorting them based on their word overlap (not sort, not full on difflib, not cosine similarity). ill do exclusion first though. inclusion is kinda implied here i just realized when i took them out of the license listing cites and removed blanks etc.

is a documentation license a software license. ok with the initial beginning inclusion words "(source|software|program|code|module|public(s+)license|ware|(w+)ware)" we managed to catch GFDL licenses and for this scope i think ill have to narrow it down to just purely source code or software targeting licenses. then again is font a piece of software. documentation might not be. font is considered a software in this scope of ours since the fonts might contain programmatic instructions how to draw and render them. documentation might be a part of a software package but is not considered software regardless.

i will call it a day to just remove creative commons licenses by exclusion string and remove GFDL licenses by manually providing a list of excluded licenses. the point of this thesis and this research is to provide how many public software licenses do we actually have so that we can understand the problem of open source and the reason why post-open source has begun to emerge. now i gotta just exclude and include (in that order) then just very lightly and cheaply remove the duplicates by checking licenses one by one with difflib

how do i exclude documentation from here do i need to manually exclude them? i can always just call it a day on validity threat and change it afterwards???? i started with

but soon realized that the point of this thesis is to provide a clearer picture of the actual problem that im trying to diminish here: the huge amount of open source licenses that exist. instead im removing everything else but the documentation license and creative commons thingy from the search string. will go through the excluded licenses once more just to check if it caught anything wrong. the big idea here is that because the open source licenses and the public licenses used are already sometimes too vague to be legally valid should be just left as is and just remove the sort of licenses that are already specifically not-software specific, i.e. documentation licenses, or at least creative commons is not considered as such.

the eye-pass of 87 excluded licenses included cal1.0 and cal-combined which seem to need to be included. made inclusions.txt to manually include them although containing the words creative commons, since those license texts were licensed themselves with creative commons. will eyeball the stage-2 shortcodes as well just to see anything i know by heart is not public software license. cc-by-sa-japanese was included so i had to make a manual exclusions.txt to manually mark licenses that are not public software licenses.

stage 2: inclusion and exclusion is done. 656 licenses left to remove potential duplicates from using ratcliff and obershelp. then just writing. literally just writing after this.

so for stage 2: i output shortcodes to .txt and full licenses to folder. from .txt i bake them to google sheets and match the sources for them. from google sheets i put them to github markdown for reader to see. note: the python script does not work on windows due to some path handling errors. i would have loved to use my desktop 1160K instead of laptops i5-8265U. ok wow it took 4101.285 seconds on my laptop. check the ss on my laptop

licenses texts were sorted using difflib and renamed index plus shortcode. then opened in vscode and if seem the same content minus noise the other one usually the order of the 5 licnse sites is removed as duplicate. vscode compare tool was used to help with larger license texts. data/database licenses popped up during the duplicate searching phase. added ad-hoc open data as an exclusion string

duplicate removal in tabs was done so that: i check the text if the n and n+1 and if they look pretty much the same i act and if the shortcodes look the sam i act.

stage 3 is removing duplicates by a feel factor from the shortcodes and from the difflib sorted contents. this is also just a one pass test.

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 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 anaylsis 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 software licenses are there in the top five software license listing cites?
- RQ2: How much is there disagreement in the shortcode names between different

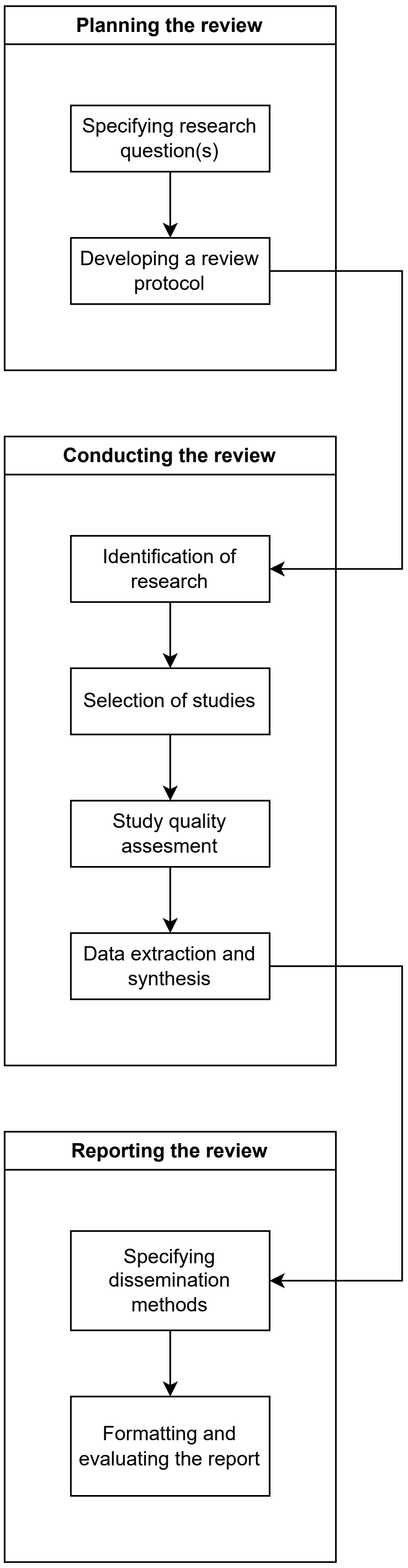
Planning the review
Specifying research question(s)
Developing a review protocol
Conducting the review
Identification of research
Selection of studies
Study quality assesment
Data extraction and synthesis
Reporting the review
Specifying dissemination methods
Formatting and evaluating the report

Figure 2.1: Three phases of a systematic literature review

public software licenses listing sites?

- RQ3: How many public licenses in software engineering does there exist?

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, per site. The review takes into account attributes like versions, supersedences to a different license family, formal or otherwise and recognizability. These attributes give us different amounts to existing PCLs in software engineering. This informatin 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 practicioners since it could give some overview and a sense of the scale when picking a public software license that would serve the practicioners' needs the best.

2.2 Search stragey

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-reproducabe results in Google Scholar and Google Search. Some PCL 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 citekitchenham2007 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 leg work 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 exclusion and quality 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

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

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 according to Balter, 2015, the MIT license (Wikipedians, 2024b). 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 PCL listing websites. This leaves us with the SPDX, Debian FSG compatibility, FSF and OSI from which all resulted in some sort of PCL 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.

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

Removing the duplicates was not intelligent and left duplicates like ZPL-2.0 and ZPL - 2.0 as unique license shortcodes. This was tackled systematically in a later chapter. The table inthe state after the strict removal of duplicates is provided in this thesis' repository (Taguchi, 2025) under the name ”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 ”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). The PCLs appear in the same order as decribed above: SPDX, DFSG, FSF, OSI and GNU. The appendix was also crafted in a spreadsheet software so that only the initial hit source was documented in the order described above. For example even if MIT license would be found on SPDX and DFSG Appendix A would only display MIT license with the ”First hit from” value being SPDX. The initial list of 789 PCLs excluding duplicates is provided in Appendix A.

Some things must be mentioned about the process of the first stage. First, the FSF outputted a ”license” named ”other”. This ”license” included at the time of observation 5282 known programs to FSF whose PCLs were not documented yet by the FSF. Although

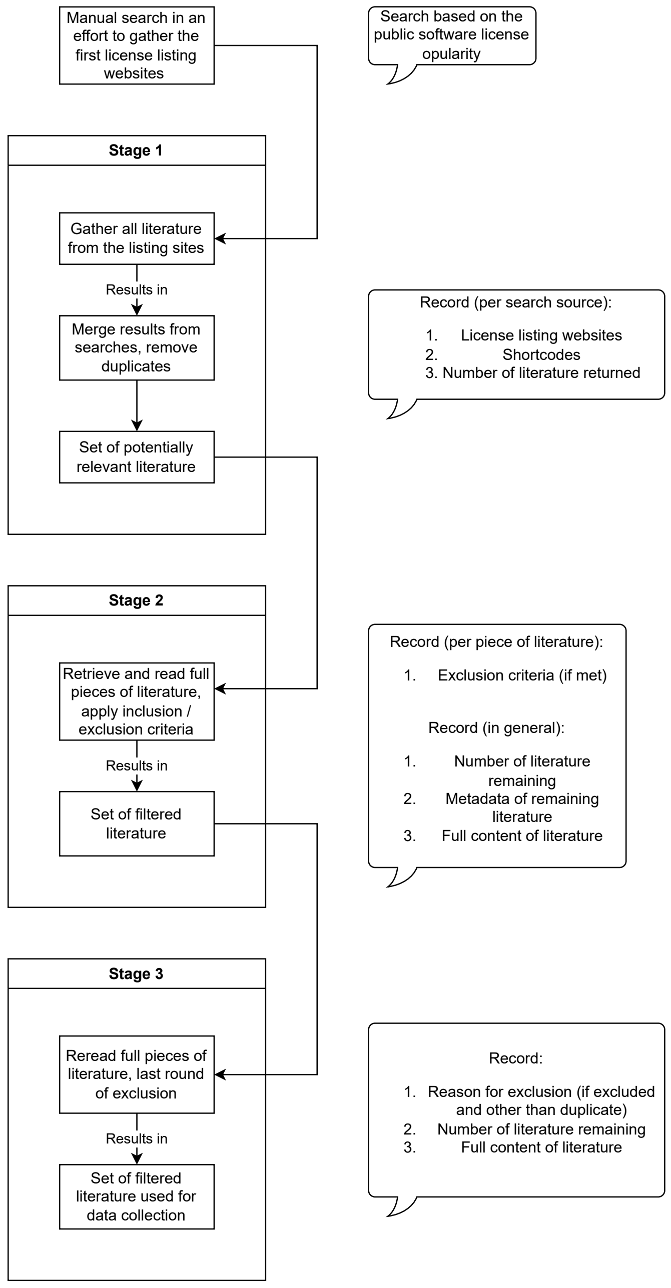
Manual search in an effort to gather the first license listing websites
Search based on the public software license opularity
Stage 1
Gather all literature from the listing sites
Results in
Merge results from searches, remove duplicates
Set of potentially relevant literature
Record (per search source):
1. License listing websites
2. Shortcodes
3. Number of literature returned
Stage 2
Retrieve and read full pieces of literature, apply inclusion / exclusion criteria
Results in
Set of filtered literature
Record (per piece of literature):
1. Exclusion criteria (if met)
Record (in general):
1. Number of literature remaining
2. Metadata of remaining literature
3. Full content of literature
Stage 3
Reread full pieces of literature, last round of exclusion
Results in
Set of filtered literature used for data collection
Record:
1. Reason for exclusion (if excluded and other than duplicate)
2. Number of literature remaining
3. Full content of literature

Figure 2.2: Search process divided into stages

some of the programs had straightforward PCLs such as GPL-2.0-only we decided to leave these PCLs out of the scope of this thesis due to the large amount of the programs. The second note is about GNU's PCLs. Since we had the most trouble scraping the identifiers automatically from this website we decided to limit the PCLs only to "Software Licenses" as defined by the table of contents on the website.

approx duplicates were the result of going to the two listing websites that had the approaximately same looking licenses. then i just checked if they were actually some sort of duplicates of one another or if they already exist somewhere else. examples here. this is also a validity threat. problem with focusing software specific licenses is for example wtfpl. it is mostly used in software licensing but it doesn't quite clearly state that it is software specific license. maybe ill have to include the word "public license" and just include stuff that's not actually software specific or maybe ill make some exclusion criteria in order to get less non-software licenses

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. This involved a manual review of the full licenses. The exclusion reason as a shortcode (e.g. I1 = failed to meet inclusion criteria 1 or E2 = met exclusion criteria 2) is provided in Appendix B.

The third stage was the most time-consuming and involved a manual review of the full licenses. 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.

2.4 Inclusion and exclusion criteria

To be eligible for the data collection and analysis, a license had to meet all of the following inclusion criteria:

- I1: The license focuses on the copyright of software source code or their binaries
- I2: inclusion criteria 2

Additionally, licenses were excluded if they met any of the following criteria:

- E1: The piece of literature is a license exception

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

Table 2.3: Data extraction form

- E2: The piece of literature is a Creative Commons license
- E3: exclusion criteria here
- E4: exclusion criteria here

The relevance of each piece of literature was evaluated based on inclusion and exclusion criteria stated above. In cases where there was doubt about the suitability of a license, a more in-depth manual examination of its content was performed. The reason for exclusion was documented for each license that failed to meet the criteria, and when it was unclear, the license was included by default.

Another relevant criteria related to the ones of inclusion and exclusion are the quality and evidence criteria. These criteria used by Dybå et al., 2007 were not put into practice in this thesis since individual PCLs per se might not be meaningful in a results, evidence nor quality perspective. This puts more emphasis on the inclusion and exclusion criteria so that is something we must be mindful about.

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 extracted data was kept for anaysis and is available as Appendix B.

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 as Appendix A, 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.

mention design science, the artifact and its contributions

how many licenses and why

statistical overview with figures (mapping study)

how many licenses during each stage (figure)

basic statistic on final licenses (figure)

essential statistics (figure)

3.1 Placeholder question (RQ1)

figures and literature identifier tables

3.2 Placeholder question (RQ2)

figures and literature identifier tables

3.3 Placeholder question (RQ3)

figures and literature identifier tables

3.4 Placeholder question (RQ4)

figures and literature identifier tables

4 Discussion

indications

follow-up observation

observation 1

observation 2

sum-up from those two

4.1 Implications for research

how to improve scientific scene 1

how to improve scientific scene 2

how to improve scientific scene 3

4.2 Implications for software engineering professionals

how to improve professional scene 1

how to improve professional scene 2

how to improve professional scene 3

overall

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 PCL lists to the Wikipedia article of the MIT license.

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 PCLs 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.

The first phase of filtering has some notable limitations starting with the two PCL 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 PCL with an identifier ”MIT”. The results of phase 1 will not include any other PCL 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 PCLs in phase 1 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 PCL 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 PCLs, duplicates and the lack of already existing tools makes this problem multilayered. However this is the integrity level we decided to live with.

FSF's PCL listing introduced us to pick another limitation for the scope of this thesis. The license shortcoded as ”other” was not a PCL 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 PCLs found indirectly from the category ”other”.

tell about the validity threats of osi literature selection for review

Lastly, 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 PCLs as software PCLs. On top of this the PCLs 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 PCLs were only the ones that were simply under the header ”Software licenses”. In the worst case scenario GNU project could have misinterpreted some PCLs 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 PCL 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 PCLs 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 PCLs in the literature selection for review but these type of duplicates are removed in phases 2 and 3 due to the PCLs being read in full.

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

4.3.2 Limitations in data extraction

importance of data extraction

lack of measurements and tooling

5 Conclusions

primary objective of this study

conclusions from each rq

5.1 Future research

adopting a clear baseline

why agplv3re is the best license

Docker CLA, SSPL

make cla easier maybe with gpg / joplin easy cla sign

LICENSE highlighting.js

what kind of efforts and why

what this thesis has provided

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.

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.

Dybå, T., Dingsøyr, T., and Hanssen, G. (Oct. 2007). "Applying Systematic Reviews to Diverse Study Types: An Experience Report". In: pp. 225–234. ISBN: 978-0-7695-2886-1. DOI: 10.1109/ESEM.2007.59.

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.

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.

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.

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.

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.

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 (2024a). *Category:PCLs — Wikipedia, The Free Encyclopedia*. https://web.archive.org/web/20240131085240/https://en.wikipedia.org/wiki/Category:Public_copyright_licenses. Accessed: 2024 Jan 31.

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

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				