Seminar Paper

Open Source Licenses

Principles and Comparisons of the GPL 3.0, LGPL 3.0,AL 2.0, CPL 1.0(EPL 1.0), and OpenJDK with ClASSPATH Exception

by

Victoria Volkova

Matriculation Number: 01247673

Supervisor: ao Univ. Prof. Dr. Rony G. Flatscher

Course: 4152 Seminar aus BIS

Semester: SS 2021

Submission Date: 01.06.2021

Erklärung:

Ich versichere:

dass ich die Seminararbeit selbstständig verfasst, andere als die angegebenen Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfe bedient habe.

dass ich dieses Seminararbeitsthema bisher weder im In- noch Ausland (einer Beurteilerin/ einem Beurteiler) in irgendeiner Form als Prüfungsarbeit vorgelegt habe.

dass diese Arbeit mit der vom Begutachter beurteilten Arbeit übereinstimmt.

Datum 01.06.2021 Unterschrift: Victoria Volkova

Table of Contents:

1. Introduction

1.1 Motivation

"Software is a unique technology, in that it comprises rights that are protectable under copyright law, patent law and trade secret law."

defines Classman in his practical guide to Software Licensing (Classman, 2016) . He also adds that in order to comprehend how a software licensing functions one has to grasp intellectual property rights "contained in the software's bundle" (Classman, 2016).

People do not realise that the majority of software is actually open source licensed (Neary, 2018). Software licensing as a topic often discomcombulates non-professionals and end-users, specifically by its terminology in regards to the definition of the word "free". Commonly, the word refers to "free of charge" which is precisely the mistake, which will be explored in this paper. By the virtue of open source licensing, many IT applications can be used today. Without open source, we could never use smartphones with their multiple softwares, nor could private companies establish their own websites. The whole Google system and its technologies have been built based on the open source softwares (Neary , 2018).

Still, not only is open source one specific type of software licensing, but the whole system of licensing computer programs is a complicated topic where a lot of confusion exists. Not everybody is an expert in software programs or legal issues, especially on the international level. There are a lot of confusing points in the terminology, conditions of use, and the number of licenses in general. Often, corporate and private users confuse free software and license with a cost-free service or product (Wikipedia, 2021).

1.2 Paper Structure

Given the purposes above, the structure of the paper is built with the purpose to enlighten on the topic step by step and partially provide explanations and terms that will be needed for understanding of the next paragraph. The paper begins with the history of the open source licensing as it explains the motivation and the values of the movement and licenses under open source terms and conditions. Trends and importance of the open source licensing will also be explained in this section. Next, in order to grasp the content of the licenses analysed in this paper and the differences between them, the most important terminology will be explained. The following component of the work is the licenses with their respective conditions: GPL 3.0, LGPL, CPL, AL 2.0 and OpenJDK with Classpath Exception. The comparison charts and suggestion of use for developers and end-users finalize the paper.

2. A Brief History of the Software Licensing

2.1 From a Printer to the Free Software Movement

The idea of sharing a program code in order to speed up a working process came to the Programmist Richard M. Stallman around the 1980s. A single improper functioning printer brought the following idea: to code an algorithm designed for a printer driver that would notify anybody who intended to use the printer that it needs some repair. It would notify the user with “the printer is jammed, please fix it” (Neray, 2018). This notification system inside MIT saved a lot of time for its employees. In 1980 MIT received a donation in the form of new laser printers. Of course, Stallman thought to connect them to the sharing system but the source code of the printer drivers was proprietary.

During the 80s and 90s, selling licenses separately from the software turned into a common practice, but the projects were exclusive and private without any opportunity for collaborations. As many software engineers working in such businesses were from the MIT University, Stallman decided to form his own and completely new operating system. His goal was to provide end-users and developers with equal basic freedoms and opportunities to understand and maintain working principles of the system. Stallman ended up founding the free-software movement (Neray, 2018). Under Stallman's guidance, the GNU operating system, which stands for “GNU’s ‘Not Unix’”, was created. It is described as a copy of Unix OS but with no restriction for end-users. Already in 1985, Stallman launched the Free Software Movement and issued The GNU Manifesto that remains to be the official document of the Free Software movement. The first version of the GNU General Public License, based on the copyleft components, was issued in 1989.

2.2 The Open Source Initiative

The history of this licensing approach also dates back to the 1990s. GPL granted the then new ability to modify a code and to share a modified software. Together with the development of the Internet, the idea of open source was drastically more popular and valuable.

The next turning point in the spread and popularity of the open source movement was the publication of the notable book by Eric Raymond called “The Cathedral and the Bazaar” in 1997 (Wikipedia, 2021). In his book, Raymond tackled the purposes and consequences of hackers’ actions and analysed and described free-software basics and its benefits. Raymond also compares the two free software models: the Cathedral Model and the Bazaar Model, which is more related to the open source licensing as a source code here is created publicly via the Internet. In the Bazaar Model, a code that is still being modified and adapted, is always available to all developers or users without restrictions or additional conditions. The author of this essay points out that with more participants in the development process it is faster to identify mistakes and bugs in the code and software.

What is interesting, is that Raymoond provides in his essay the list of 19 lessons from the past software developments. Some of them clearly illustrate his affinity to open source ideology, for example:

“*If you lose interest in a program, your last duty to it is to hand it off to a competent successor.”*

“Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging.”

“The next best thing is to have good ideas is recognising good ideas from your users. Sometimes the latter is better”

The book motivated Netscape Communications Corporation to present its Internet Suite as a free source code, free software. 1998, the Open Source Initiative, or OSI, as the next major historical event in software licensing, was founded. The idea behind the establishing of the official Open Source Software organisation was to avoid the association with something one can get for free or steal someone's intellectual property. That way, the term free-source software was replaced with the new term, used till nowadays, open-source software, shortly OSS. The founders of the Open Source Initiative, Eric Raymond and Bruce Perens, stressed transparency, affordability,security and flexibility as the main advantages of publishing a software with an open-source code (Tozzi, 2016).

2.3 Open Source Definitions and Characteristics

The OSI’s definition of an open source is internationally acknowledged as a standard official definition. The definition is based on the Debian Free Software Guidelines (DFSG) written by Bruce Perens. There are 9 key guidelines a free-software must obey:

- Free redistribution.
- Inclusion of source code.
- Allowing for modifications and derived works.
- Integrity of the author's source code (as a compromise).
- No discrimination against persons or groups.
- No discrimination against fields of endeavour, like commercial use.

- The license needs to apply to all to whom the program is redistributed.
- License must not be specific to a product.
- License must not restrict other software.

In other words, an open source license according to Perens provides a source code of a software for common use without restrictions. OSI uses this definition for analysis and approval of a software as an open source. The successful examples of software that were approved by OSI are such worldwide known projects as Linux and Mozilla Foundation supported the organisation and are also open-source-software.

Gregorio Robles also complemented the OSI's definition with the next conditions:

- In a valid open source software users are also co-developers regardless of the skills and scope of action: every user has to be able to access a source code. The idea of an open-source community is also to stimulate the development environment and to encourage users to add their code modifications and report any bugs occurring.
- It is crucial to release software as soon as possible to involve more programmers.
- Integration of bug fixes must be done so often as possible, sometimes even automatically.
- To allow more developers to work at the same time on the code, software has to be of a high modularisation.

2.4 Importance of the Open Source Software Licensing

It is now common for software to be licensed, rather than sold to enforce compliance of the user with terms on which they can operate the software. This could include the time period service, how many devices can adopt the software, what types of application is appropriate (commercial or educational) and possible restrictions on reverse engineering i.e. extracting existing design, ownership rights i.e. selling and transferring which are normally mentioned in the Software License Agreement. When such agreement is violated, the user can potentially lose the right of use and be eventually fined. (Writer, 2021)

The choice of software licensing is both strategically significant for big companies, established corporations, or small start-ups as every business model today at least partially considers the digitalisation of a company's structure. Furthermore, there are a whole set of risks connected with adopting unlicensed software, for example: pirate software can contain viruses, let a computer download some malware once connected to the Internet or just make the work with a partner company that uses licensed software impossible.

According to the Apache Foundation, using open source software models and licensing contributes to the success of businesses, saves customers up to 60 million dollars yearly according to the Standish Group Report 2008 and improves the wellbeing of the society in general. Open source models have a significant impact on the development of the IT Industry. Meanwhile, it brings as well some personal benefits such as a sense of membership and a feeling of dignity while working on the worldwide used and known complex projects (Jagielski, 2015).

2.5 Future Trends for Software Licensing

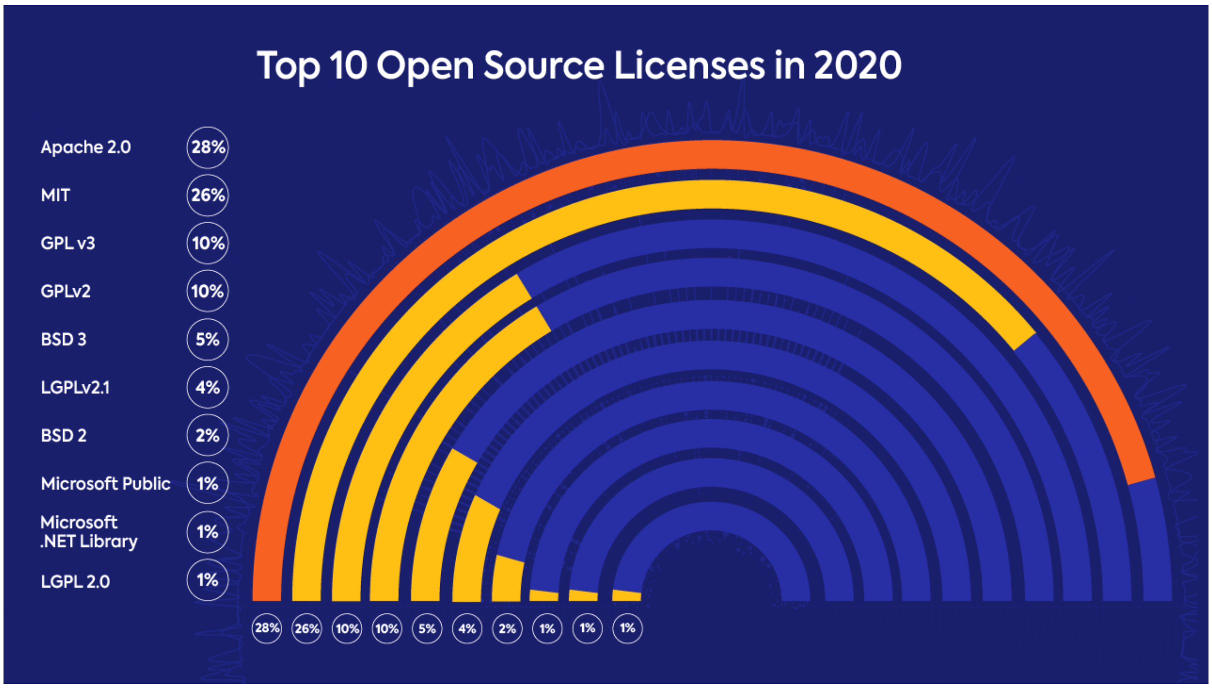


Image 1: "Top 10 Open Source Licenses in 2020." The image is taken from https://www.whitesourcesoftware.com/resources/blog/open-source-licenses-trends-and-predictions/

The most popular and most widely applicable open-source software licenses have been Apache Licenses so far after MIT License and GPL 3.0 and this trend for Apache will be more vivid in the future (Johnson, 2021). Apache is adopted by many worldwide projects such as Swift, PDFj.s and Kubernets (InfoTechNews, 2020). Only Google claimed that 25% of projects on Google code use Apache Licensing, for example, the Android Operating System (Wikipedia, 2021). Anyway, larger Corporations, such as Microsoft, have been arguing over time on how ethical and fair unpaid creations, outputs and innovations are. Expectedly, fresh maintainers and software developers continue to join the community, as there is a rising amount of small businesses adopting open-source forms. The popularity of open-source communities is explained by the convenience of the cloud solutions for big corporations and for small entities where open source licensing is a base for functioning.

Furthermore, open licensing attracts businesses with the opportunity to protect themselves legally and cooperate with other private companies and start-ups for synergy effects (InfoTechNews, 2020). The practice shows: “less restrictions, the better” according to Johnson, 2021. Huge open source communities will always come up with new ideas and opinions on the most effective online business models, but the demand for the open source program has never been as huge as now (Johnson, 2021).

3. Definitions

3.1 Software License

To begin with, a software license is a contractual agreement based on the copyright law and is the only way to determine a software product and control its usage, how and how often, for what purpose, for how long and by whom and spread. Software licenses protect basically a unique source code of every software that is someone's intellectual property similar to a written article or a piece of music (Gilbert-Knight, 2012). Such licensing contracts can regulate an access to underlying licensing codes as it is possible to use more than one license at the same time, and provide end users with a possibilities to copy the licensed program, modify it or even redistribute. Software licenses are also important in terms of international use, the privacy regulations and sales restrictions in different countries.

3.2 Source Code

Codes divide themselves into 2 categories: source code and object code. For the software licensing source code is relevant. Source code is a human-readable text that supports and speeds up programming and development of new software. Source codes are an intellectual property, the usage of which is protected or managed by licensing. For understanding the principles of work of every software license it is crucial to know what happens technically and for what actions he or she is responsible for while using a software. Furthermore, binary coding, which can be executed by the hardware, is crucial for software redistribution that is why source code is a necessary requirement for the open source licensing. (Wallask, 2019)

3.3 Free and Non-Free Licenses

There are four essential freedoms or rights that ensure the protection of liberty like protection of free speech when it comes to free licenses:

- The freedom to run the program as a private user or a commercial organisation wish, for any purpose (freedom 0). Furthermore, end users have the right under the free source licenses to share or distribute the program to other end users who become this freedom automatically. The central figure in the free software license is a user not a developer.
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this. In the other words, it is possible to run the program under an own modified code instead of the original code. Indeed, if any change represents an improvement of a software is very subjective and hard to access, That is why when a license states that only changes, but not improvements, are allowed, the program is not free.
- The freedom to redistribute copies, original or modified software, free or against a charge for sale (freedom 2). “Free” means here to have the right to share a software without paying for sharing any fee and with no need to ask for a permission from developers.
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. In this case the license cannot be with copyleft. Access to the source code is a precondition for this. A program has to contain (Free Software Foundation, 2021).

“‘Free software’ does not mean ‘non-commercial’. On the contrary, a free program must be available for commercial use, commercial development, and commercial distribution. This policy is of fundamental importance—without this, free software could not achieve its aims.” (Free Software Foundation, 2021). This highlights the entire philosophy of this concept: sharing the GNU system with everyone, like businesses and individuals. Hence, requiring commercialisation. This is important for communal success without any obstructions. That being said, a common confusion surrounding free software is the belief that it must mean *payment-free.* In reality, it reflects on the freedom the user is granted by adhering to the terms.

A free license must always comply with the four freedoms to be considered a free program. Setting boundaries, eliminating some freedoms, and requiring users to pay is commensurate with exclusion and therefore acts as non-free.

The core issue of the free software licenses is not in its price but in the freedom and justice. Analogue to the freedom of speech, the philosophy of Free-Software and the Free Software licensing respects the freedom of any end user to use a software, to copy it, to share, to explore and to change or even improve it. Some institutions name free software in French Libre Software to avoid the association with free of cost software and licenses. The principle of freedom is protected by the rights of every single end user or the whole community of users to control the programs, conditions and the way it functions. (Free Software Foundation, 2021)

Besides, free software is not equivalent to non-commercial software. Thus, a free program must be valid for commercial purposes including usage, development, and trade with the goal to make profit out of it. The way a user got the program and license, by purchasing, copying or simply downloading it for free, this user still has these four basic freedoms.

3.4 Open Source License

While open source and free software stand for similar array of programs, they follow different ideologies. As stated above, free software pushes for freedom and justice. In contrast, open source sees more of an importance in feasible advantages rather than morals. Which is why many people choose to not use it, and not even use the term. In a practical sense, open source's criteria is looser and licenses are more restrictive than those of free software. Additionally, its source code does not carry a copy left, making the license weak. They often also include supplementary non-free conditions as IMicrosoft and its Visual Studio Code. (Stallman, 2021)

“Many products containing computers check signatures on their executable programs to block users from installing different executables; only one privileged company can make executables that can run in the device or can access its full capabilities" (Stallman, 2021). Such devices are often referred to as *tyrants* and the discipline is called *tivolization.* This is usually looked down upon as executables may be made from free source code and carry a free license but users cannot run or modify versions of it. In practice, making it non-free.

There is a lot of confusion and stretching of the truth when it comes to open source licensing. The official definition incluses 10 components:

1. Free Redistribution
2. Source Code
3. Derived Works
4. Integrity of the Author’s Source Code
5. No Discrimination Against Persons or Groups
6. No Discrimination Against Fields of Endeavour
7. Distribution of License
8. License Must Not Be Specific to a Product
9. License Must Not Restrict Other Software
10. License Must Be technology Neutral

(Open Source Definition, 2021)

Thus, open source is more than just giving access to a code. Free redistribution, however, means that a license under open source conditions must not demand any fees, royalties or restrict any member or user to give away or even sell a software. Concerning a source code, it has to be included in a program which explicitly allows distributing it. Another requirement is that it has to be simple and moldable for other users to be able to modify it as well. Along with a typical open source, license must authorize any changes made to a code and derived works. All modifications have to be distributed under the terms of the license, under which an original software was licensed. This facilitates experiments, lures new talented minds and catalyzes evolution. The principle of the Author Source Code is that the license must explicitly, without any extra notices or conditions, permit a distribution of a “software built from modified source code” (Wikipedia, 2021). In that case, diverse names or version numbers, in order to identify the modified software, are allowed. That is because authors and users have the same rights. Both stakeholders have the possibility to search on who modified a code, who is responsible for the software program, and what they support while using this particular version.

For the worldwide community to function and serve its primary goals, some further rules are evidently needed. When it refers to communication and co-working, an exchange of information and coding implies that every single user is treated equally and has equal rights and possibilities. Open Source Movement suggests no discrimination against Persons or Groups and Fields of Endeavour. While there must not be any restrictions in the software license, a country may have some laws or regulations forbidding usage of some software or applications on its territory, or vice versa an export of software would be restricted, so that has to be separately clarified. The component “No discrimination against Field of Endeavour” allows end users to use a specific software for any purpose he or she would need it: analytics, social research, statistics, creating new programs, practice, study or building a business model. (Open Source Initiative, 2021)

Finally, a license must be ubiquitous so that it is applicable to all kinds of products, devices and does not restrict or impact other licenses. Open Source licenses must not affect with any restrictions other software installed on the device. Other programs on the device can be closed-sourced, or restricted in use, because it will bother the work of commercial users that use primarily programs and applications with the closed source codes (Winslow, 2019). Open source licenses also have to give a possibility to share or redistribute programs offline, not over Web channels, so without click-wrapping or downloading.

In conclusion, “open source is a licensing model, but also a development model. The license is the step that enables the IT-industry development.” (Heather Meeker, 2020).

4. GPL 3.0 - General Public License

4.1 Basics

Written by Richard Stallman, GPL or GNU GPL is one of the most frequently used *free software* licenses. Users are able to use, modify, and distribute it freely. The fundamental facet of the GPL is copyleft. While being a play on word copyright, it serves an opposite purpose. It applies a copyright law, where all the modified or derivative versions must be issued under the same license with the same protection. It employs the 4 freedoms mentioned above to protect the rights of development and distribution (*see 2a*).

The developers rely on these 2 steps to protect users rights: “(1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it” (4.1) There is no warranty for authors or users, however it is required for the modified version to be marked as such, so the problems will not be attributed to the previous owners. However, there are devices that are developed to refuse the access to instalment or launch modified versions of the already established software inside them. Therefore, this version is designed to forego such abuse of initial freedom and will continually be modified in order to provide the users’ freedom in extension to those domains.

4.2 Terms and Conditions:

1. Source Code
2. Basic Permissions
3. Protecting User's Legal Rights From Anti-Circumvention Law
4. Conveying Verbatim Copies
5. Conveying Modified Source Versions
6. Conveying Non-Source Forms
7. Additional Terms
8. Termination
9. Acceptance Not Required for Having Copies
10. Automatic Licensing of Downstream Recipients
11. Patients
12. No Surrender of Others' Freedom
13. Use with the GNU Affero General Public License
14. Revised Versions of this License
15. Disclaimer of Warranty
16. Limitation of Liability

4.3 Permission and Limitation of Use:

Any software under GPL can be run for all purposes, commercial or not. Users can be charged for copies or provide them gratis. For individual or personal use, the codes can be modified without having to release it. For distribution, the entirety of the code has to be published for the end-users, including all changes and extensions. This is where copyleft is employed to protect freedoms.

GNU assures that a program consisting of original source code, or even when it is combined with source code from other software components, does not need to be licensed under GPL and does not need to be published; even though their underlying system is licensed under GPL, meaning anything running on it, are not considered to be derivative works (Smith, 2014). If the GPLed parts are being utilized in a program and shared, then all the source code has to be published and managed under the same license.

Additionally, Article 11 of the WIPO Copyright Treaty states that users cannot be held accountable for bypassing digital rights management (DRM) nor is GPL-licensed code an adequate technical protective measure. (Smith, 2014)

In the end, the “goal in designing the GPL was to maximize its use of the controls of copyright law to maximize the number of works that were covered by GPL” (GNU General Public License, 2020)

4.4 GPL 3.0 Notice

One can apply GPL 3.0 terms to a program by attaching a license notice to a program, more precisely, to a start of source files that has to contain a warranty and liabilities disclaimers, a copyright line and link to the full version of a license.

Any license notice must include the next text:

<one line to give the program’s name and a brief idea of what it does>

Copyright © <YEAR> <NAME OF AUTHOR>

This program is free software: you can redistribute it or/and modify

It is under the terms of the GNU GPL as published by the Free Software Foundation, either version 3 of the License, or (at your opinion) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY;

Without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. Iff not, see <http://ww.gnu.org/lienses/>

It is recommended to also add contact data, for example, an email so that other developers or users could easily reach the creator.

Developers have to think ahead on what they want to do with a program in the future and on the purposes the other people would use it. GPLs do not permit any integration of a program that was licensed under the GPL terms with the proprietary programs. In some cases then the GNU LGPL is more functional and effective. The next chapter concentrates on the LGPL and clarifies the differences between the GPL 3.0 and LGPL.

5. LGPL 3.0

5.1 Basics

The GNU Lesser General Public License Version 3 was developed in 2007 with a weaker copyleft system than the one of GPL, meaning it does not require personal custom source code (not including LGPLed parts) to be published under the same license terms. It was developed as a concession of the strict copyleft of the GNU GPL and more relaxed licenses of the BSD Licenses and the MIT License. It is still, however, compatible with GPL. The language behind the word “lesser” was carefully chosen to indicate the possibility of freedom restriction in the end-user’s handling of said software. Essentially, the user is given the freedom only for modification of LGPL licensed components, excluding any proprietary ones. In the case of proprietary software, LGPL codes are commonly operated in the shared library format, so there is a clear difference between proprietary and LGPL components, which is what LGPL is most commonly used for. (GNU General Public License, 2016)

*5.2 Terms and Condition*s

This particular license allows users to incorporate LGPL software components into their personal work (even proprietary) without being obliged to publish their source code by the copyleft. This version of the GNU Lesser General Public License includes the terms and conditions of the GNU 3.0, such as having to publish their modified version under the same LGPL license, as well as 6 new supplementary additions, such as:

“*2. Conveying Modified Versions.*

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) Under the GNU GPL, with none of the additional permissions of this License applicable to that copy. //

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

b) Give prominent notice to the combined library that part of it is a work based on the Library, and explain where to find the accompanying not combined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns." (GNU Lesser General Public License,)

The main advantage and the initial purpose of Lesser GPL, as stated above, is compatibility with various software libraries. "Using the ordinary GPL is not advantageous for every library. There are reasons that can make it better to use the Lesser GPL in certain cases. The most common case is when a free library's features are readily available for proprietary software through other libraries. In that case, the library cannot give free software any particular advantage, so it is better to use the Lesser GPL for that library." (Free Software Foundation, 2016)

The first alterations in approaches to operating system services started with the GNU C Library. It has now developed into a drop-in replacement for previously existing C Libraries. Meaning, that one can switch one "component with another one without any other code or configuration changes being required and resulting in no negative impacts" (Wikipedia,

2021) The entire idea based on the already introduced ideology of inclusivity and freedom. With that being said, the C Library provided adequate and equivalent service and functionality in comparison with other C Libraries under a Free Software-operating system-umbrella.

5.3 C Library for the Free Software

The licensing of *glibc*, the LGPL related GNU C Library, differs. A lot of applications depend on the C Library, meaning they are obliged to link the library to operate the system. Additionally, all *Unix-like* (a complete operating system package with several different components that formed a self-contained software infrastructure) systems must be incorporated with the C Library to create a new product comprised from the original work and the C Library. Hence, if glibc is under the GPL licensing umbrella, precisely every application distributed for use on GNU or Linux would also need to be licensed under GPL, because to even launch or operate they have to be incorporated with some larger products through linking with glibc. Even though the primary intent of developing glibc under the GPL was to protect GNU or Linux from proprietary applications, it would be virtually impossible. On this basis, the Lesser GPL was conceived. (Free Software Foundation, 2021)

6. AL 2.0 - Apache License Version 2.0

6.1 The Apache Software Foundation and Apache's Philosophy

The Apache License is a license that was created under the prerequisites of and approved by the Apache Software Foundation. For a better understanding of the idea behind these software license agreements and strategies it is crucial to study the ideology of the Apache Licenses and Products creators.

The Apache Software Foundation, or the ASF, was founded in 1999 as a non-profit organisation. Till now, it has been a virtual organisation of individuals that wish to volunteer and contribute to the progress of the community as the ASF has not paid its volunteers for working on any of the Apache projects (Jagielski, 2015). Apache Software Foundation exists in order to provide legal, organisational and financial entities, and support for the various open source software, or OSS, products.

Jim Jagielski believes in the motto “Let the developers develop”. Jegielski is the core developer of the Apache HTTP Server and a co-founder of the ASF. In his ASF presentation, Jagielski also adds that “the Apache way is about how everything cooperates.” The main mission of the Apache creators is to provide an open-source software to the public, make it available for free of charge, as well as to provide a foundation for collaborative and open software development projects by supplying hardware, communication, and business infrastructure. In that sense, the ASF is an independent entity whose mission is to supply individuals with needed equipment, donate resources, and be assured that these donations will be used for the public benefits.

Jagielski also positions the ASF as an organisation that provides support for the whole Apache community, beginning with its outstanding software developers, contributors, and to the end users in working on the open source software projects under Apache conditions. One of the goals of the organisation is to contribute to the healthy competition of companies in different industries or their collaborations. Apache Software Foundation focuses on its community and explains the success of the Apache license code by the high integrity of each member. “If we focus on our community, which grows and educates together, the good will be exponential. There will be a natural air in it,” says Jagielski in his Apache presentation. The official vision statement of the ASF sounds as follows: “If the code was created by our community, our code should be exceptional”.

The Apache Projects are characterised by collaborative, consensus-based development processes, an open and pragmatic software license, and a desire to create high quality software that leads the way in its field. The ASF motivates its community to work with enthusiasm, exercise together and learn from each other. The organisation wants to profit from its open source format. That way, industries, market professionals, software developers, volunteers, service providers, and end users work together, share their ideas, concerns, views, and experiences with each other and create a license that all desire. The Apache Software Foundations and its founders believe that software developers and programmers are artists, who constantly put efforts, try, innovate, change their minds and never feel satisfied with their work (Jagielski, 2015).

The Apache License Version 2.0 was also created under peer review in a public and transparent way. Jim Jagielski explains the choice to make the license open source by the possibility for all users, independent from its role in the community, to access the code, avoid vendor lock-in, and achieve more nimble development and frequent releases. He admits that open source licenses make lawyers concerned on how to rule crossing restrictions. Anyway, Jagielski is proud of their community.

6.2 The Principals of the Apache License Version 2.0

Speaking about the license itself, Apache 2.0 is a very permissive license that is community and vendor friendly. According to Jagielski's presentation, the most fast growing open source license, which exists today. He also claims that the Open Standards work better with the “AL-like licenses” due to its wide adoption.

The work of the Apache License is based on the following principles: mutual trust and respect, collaboration, consensus decision making, responsible oversight, individual participation, peer-based work, meritocracy and do-acracy (Jagielski, Apache Foundation, 2015). If mutual trust and respect are clear in terms of meaning, meritocracy is interpreted as a qualitative and solid contribution to the society. The desire to input knowledge and skills into the open source project has to be governed by “merit”. Those who have more merit in the projects, get more responsibilities and motivate other users and contributors to do so. “Merit never expires” (Jagielski, 2015). Next, the do-acracy regime rules the Apache community: the more a user does, the more he is able to do. If a simple user suddenly starts to participate in researches or other activities that are dedicated to the license evaluation and promotion he becomes a committer, then a member of developers, a project member and the most serious engagement level is a foundation member in some of the Apache’s projects.

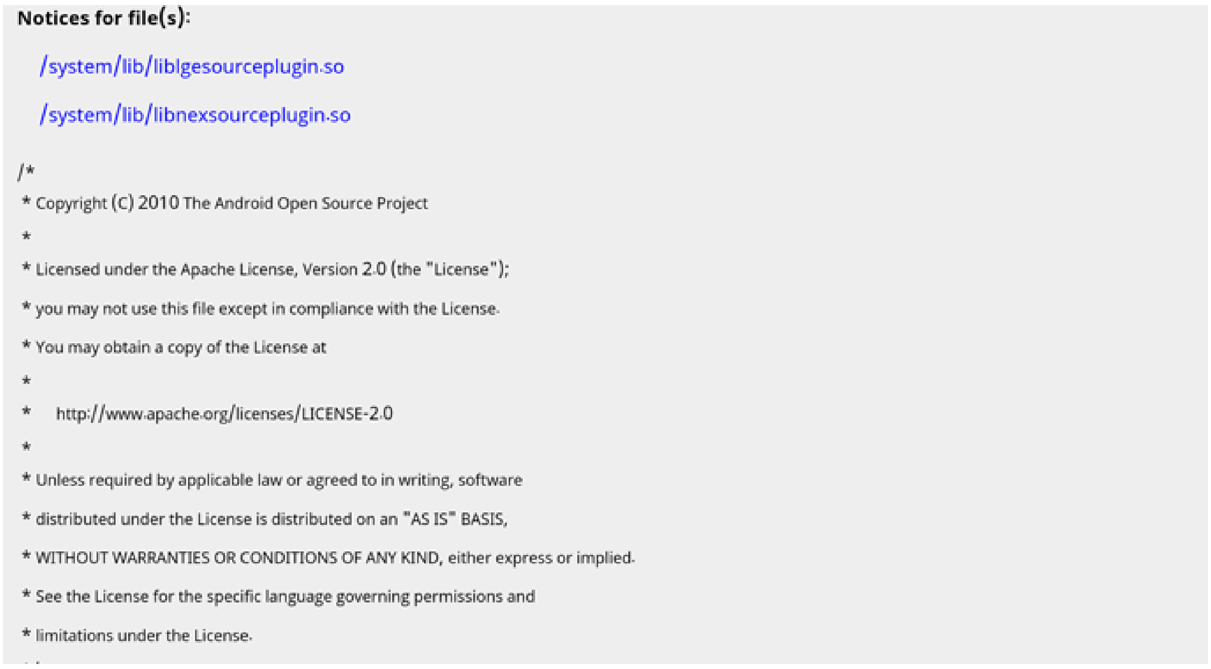
But the most important rule that explains Apache’ success is that here every single developer is regarded as an artist and individual. Every developer holds exactly one vote. Everybody has equal weight of their vote and can participate in decisions by simply voting. This system is fair enough and allows to avoid stagnation in the community as everybody naturally involved in the decision making process. It proves that Apache Foundation is based on the community created code, collaboration of diverse programmers and scientists, and consensus.

Jim Jagliesky explains the term “community code” by the way people volunteer their resources to the society’s well-being: time, interests, experience, money and efforts. Since there are no requirements on skills or knowledge in order to participate and work on the projects, the Apache community is very diverse: “End result - better long term code”-states Jim Jagliesky in his presentation. Speaking of the software development process itself, it happens on-list, online and at least 3 active developers are required in order for a project to be valid.

6.3 Distinctive Characteristics of Apache License 2.0

Apache License Version 2.0 was approved and registered by the ASF in 2004 and refers to the permissive types of Open Source Licenses. According to Github.com, 1.248.000 Projects have been created under this license.

In order to apply AL 2.0 to the work, program or a separate file, one just has to include a copy of the license and a notice file with all references to it in the work.



Notices for file(s):

/system/lib/liblgesourceplugin.so

/system/lib/libnexsourceplugin.so

```
/*
 * Copyright (C) 2010 The Android Open Source Project
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
```

Image 2: "Apache License Notice" The image is taken from https://www.apache.org/licenses/LICENSE-2.0.html

According to the Apache License Version 2.0, as it is a community code, every contributor automatically grants any successor user the "perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source and Object form" (Apache Software Foundation; 2004) - a grant of the copyright license.

As concerns a grant of the Patent License, every contributor according to the AL 2.0 license terms grants any subsequent recipient or user "a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable patent license". Recipients of patent license have equal rights to use and even sell a licensed program, software, component or application.

Separate set of rules and conditions are dedicated to the redistribution of the software AL 2.0-licensed. In order to legally redistribute a software, its copies and derivative works, user has to fulfil the following criteria (Apache Software Foundation, 2004):

1. User has to share the next recipients of software or its derivative work with the copy of AL 2.0 License
2. Every single file has to include some notice on a user's changes to the code or any file; in these license notice, it has to be stated explicitly who conducted changes. This is the single restriction for the developers under the Apache 2.0.
3. Contributors must retain in the source form of derivative works that are meant to be distributed "all copyright, patent, trademark, and attribution notices from the source form of the work".
4. Every copy of the work or derivative works must also include a license notice copy if a contributor intends to redistribute software and derivative works. Contributors can add their own attributes or text to the official license notice. It is only not possible to change already existing text in the license provided by the ASF.

Thus, the Apache license is a balance between end-users and creators. It does not permit to change any attribution notices, trademarks, patents, and copyright that already exist in the work source code. Nonetheless, they can share, modify, download, and execute a program for any purpose without any additional legal steps. So, Apache 2.0 does not impose any restrictions on its users making software licensing especially popular. The next significant point for businesses is that Apache stimulates commercial use and does not require companies to reveal technologies or skills used in order to construct a program on the basis of the work code (Mazur, 2017).

To sum up, the Apache License Version 2.0 is a permissive license, which unlike the other free and open-source licenses allows the distribution of software and its modifications under a different license from the Apache 2.0 license. Though it is still strictly required to add the original license and a license notice to all unchanged and unmodified programs besides the original documents such patents, copyrights, notice and trademarks. The license is very safe in terms of using patents, as every user is assigned his or her license for every patent. In return for that grant, the user loses his or her license and all rights if he takes legal action because of the patent infringement.

7. CPL 1.0 - Common Public License

7.1 Basics

The Common Public License, as its name already suggests, is an open-source and free software license. CPL was published by the American technology company in May, 2001 and was approved by OSI and FSF (Wikipedia 2021).

The terms of conditions of this license allows end users to use software licensed under CPL with products licensed under proprietary terms or other licenses. For instance, the Eclipse Public License was created on the basis of the CPL conditions but was modified with extra clauses to be transferred into a more restrictive license (Wikipedia 2021).

The CPL was created with the intention to build up a compromise between the developers' rights and the users' freedoms. This license maintains and stimulates synergetic open-source development and at the same time gives an exclusive opportunity to developers to practice and run software CPL- licensed and licensed under other licenses together, and also add some components or content that are not CPL- licensed but are proprietary (Wikipedia, 2021).

7.2 Commercial distribution of software under the CPL

The main object of protection in commercial distribution are end users and business partners. Any commercial contributor has to assure that his or her software or application is included into a "commercial product offering" and his or her licensing conditions do not establish any additional or different from the CPL liabilities for buyers, end users and partners.

Besides, a commercial contributor in commercial licensing under the CPL there are so called "indemnified contributors". This classification was established to help contributors help each other to protect themselves collectively from any arising from possible legal actions losses with the third parties. Commercial contributors, when offering his or her product for commercialisation under the CPL, automatically admits that he or she is obliged to protect and indemnify other contributors. In order to be qualified as an indemnified contributor, one must:

1. notify the Commercial Contributor
2. allow the Commercial Contributor to cooperate and take part in mitigations connected with legal actions, claims and other negotiations

7.3 Disclaimers of Warranty and Liability

As the license text states, there are no warranties and limitations as each user is only responsible for concrete purposes of usage and distribution of a program. Recipients have to count with not limited risks of program errors, its costs, data losses, and other possible issues arising from using the CPL.

In the same manner it is claimed in the license text that no one, neither recipients nor contributors, has no liabilities for any kind of damages or errors as well as profit losses from using a software or application under the CPL.

7.4 Comparing CPL with GPL and LGPL

Common Public License, if to compare with General Public License, has both similarities and distinctions. For instance, CPL and GPL one and the other require the same conditions for a distribution of modified copies of software. Under CPL developers have to come forward with available to every user a source code of a modified by him or her program or a component.

That also means that similarly to the GNU LGPL, libraries licensed under the Common Public License can be linked to the non-CPL licensed programs. Any licensee can link such libraries with any software with no linked source code or any permission request.

The differences are based on the CPL license notice and requirements and grant enclosed into the body of license. Users accept the CPL license agreement just by firstly employing a program, reproducing its code or sharing a program with others (OSI Website, 2020).

Concerning the grant of rights, CPL foresees as GPL and LGPL an execution and distribution of a Work or Derivative Works made by a contributor without any payment or fee, additional obligations. A contributor can also sublicense contributions to the work, display and perform them publicly. Sublicensing is possible in both source and object code form.

Right at the beginning of the license text it is stated that the usage, reproduction or distribution of a software is equal to the agreement with the terms and conditions of the Common public license. CPL also defines what is understood under a contribution and divides contributors into two groups: initial contributors who created the program and subsequent contributors who lately brought their modifications to its program. What concerns a contribution, the CPL defines it as a distribution of the original program under the CPL Agreement by original contributors and the distribution of program modifications and extensions by subsequent contributors (Eclipse Foundation,2021).

“A non-exclusive, worldwide, royalty-free” copyright and patent license under Licensed Patents is granted for usage, sale, modification, import of a contribution in both source and object code form (Eclipse Foundation, 2021). That does not mean that a hardware is automatically licensed under these conditions.

In order to distribute a program in object code form under own license agreement, it has to:

- comply with the terms and conditions of the CPL Agreement
- disclaim all warranties and conditions on behalf of contributors concerning as well merchandising
- clearly exclude all liabilities on behalf contributors “for damages, including direct, indirect, special, incidental and consequential damages such as loss of profits”
- notify that there is a source form option and the way a recipient can get it in an appropriate manner

In order to distribute a program in a source code form, a contributor has to verify its availability under the CPL Agreement and include a copy of this agreement with each copy of a program.

7.5 The Eclipse Public License

The Eclipse Public License, or shortly EPL, is also a free and open-source software license, approved by the OSI and the FSF that today successfully substitutes the CPL. The latest EPL version 2.0 was issued on 24th August, 2017 (Milinkovich, 2019). The CPL is more suitable for private businesses especially for international practice as the choice of law provision was abandoned. The recipient can also choose either to distribute a software under the EPL license he or she initially got it or choose another one (Open Source Initiative).

Under the EPL were released the following projects: Graphviz, JUnit, OpenDaylight Project, AT&T, KornShell, Gojure Skript (Retrieved from GitHub).

8. OpenJDK (Java Development Kit) with Classpath Exception

8.1 Software Development Kit

A Software Development Kit, or SDK, is a set of various software development tools that is installed all at the same time in one package to a computer or similar device (Wikipedia, 2021). There is usually a compiler, debugger, and some software framework included in the installation package. The main function of such development kits is just smoothing the production of any applications but sometimes more advanced SDKs can contain supplementary attributes for software and app analytics, advertisements etc.

8.2 Java Development Kit

The Java Development Kit is also a free and open source application of the software development kit mentioned above. It is the most popular SDK today and similarly represents a collection of instruments that helps to enforce the buildout of Java applications, as it was in the case of Android Application on the Java Platform. Accordingly, hardware platforms are often attached to operating systems, so in order to create some app for IOS is the IOS SDK correspondingly required. JDK is basically a binary product constructed specially for Java developers on Windows, Solaris, macOS and Linux operating systems.

The JDK is only one of three “technology packages” , as the image 3 shows, used by the Java programmers. There are three components, which are:

- Java Virtual Machine that executes programs
- Java Runtime Environment that constructs the Java Virtual Machine
- Java Development Kit with the help of which Java programmers construct programs executed by both JVM and JRE.

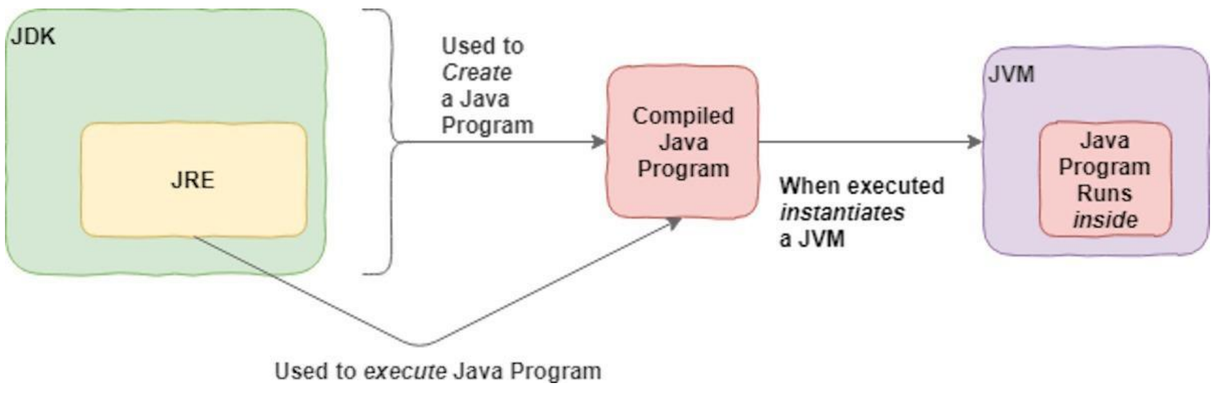

JDK
JRE
Used to *Create* a Java Program
Compiled Java Program
When executed *instantiates* a JVM
JVM
Java Program Runs *inside*
Used to *execute* Java Program

Image 3: “JDK - one of the three technology packages” The Image is taken from https://www.infoworld.com/article/3296360/what-is-the-jdk-introduction-to-the-java-development-kit.html

JDK is invented on the basis of the Java compiler and class libraries. Users can download the JDK free of any charge and create their own Java- based programs or applications. The Java Development Kit consists primarily of the programming "tools for developing, debugging, and monitoring Java applications".

The JDK was issued in many versions in order to be available for more platforms but each version includes Java standard technical components and a collection of programming tools, including the next core Java specifications:

1. **javac**: the Java compiler. It converts source codes into **Java Bytecode**

2. **java**: the loader and interpreter for Java applications. Java is needed for the interpretation of created by java compilers
3. **javap**: the disassembler of the java class file, or a file with .class extension based on the Java Bytecode in order to be executed in the Java Virtual Machine.
4. **javadoc:** tool in order to generate automatically documentation from source code
5. **appletviewer**: without a web browser, it runs and debugs Java applets
6. **jar**: the Java archiver. It classifies the files according to the class libraries and puts them together in appropriate JAR files.
7. **jarsigner**: verification and signing tools for Jar files
8. **extcheck**: Jar file conflicts detecting tool
9. **jabswitch**: the Java Access Bridge that "exposes assistive technologies on Microsoft Windows Systems".
10. **jdb**: the Java debugger

8.3 Classpath Exception

The classpath exception, or GNU GPL linking exception, permits linking a GPL library with an independent module ("which is not derived from or based on the library"), without subjecting the resulting program to the GPL and its terms. The independent module can obviously be someone's own proprietary program. Therefore, the classpath exception enables programmers and end users to implement GPL'ed license components in a certain way without risking the integrity of the Intellectual Property.

The linking as a function refers mainly to the linking of the codes in order to integrate documents in one (Java) file and in a readable and consistent format. This feature allows linking programs ,which are registered under the terms of not compatible with the CPL software licenses, to the CPL library code. The GNU Classpath linking exception is GNU GPL-licensed where the terms of the GPL explicitly permit its copyright holders to link the library with independent modules that are licensed under other terms as it is readable on the Image. Independent modules are elements based not on the GNU Classpath library. Two conditions must be fulfilled in order to comply with the GPL regulations regarding the classpath linking:

1. All independent modules or the result of linking the library with independent modules are free to copy and distribute in the case users or developers meet the requirements of licensing agreements of these independent modules.

2. If users or developers do not intend to modify the GNU Classpath library, they have to delete the exception statement from their library version.

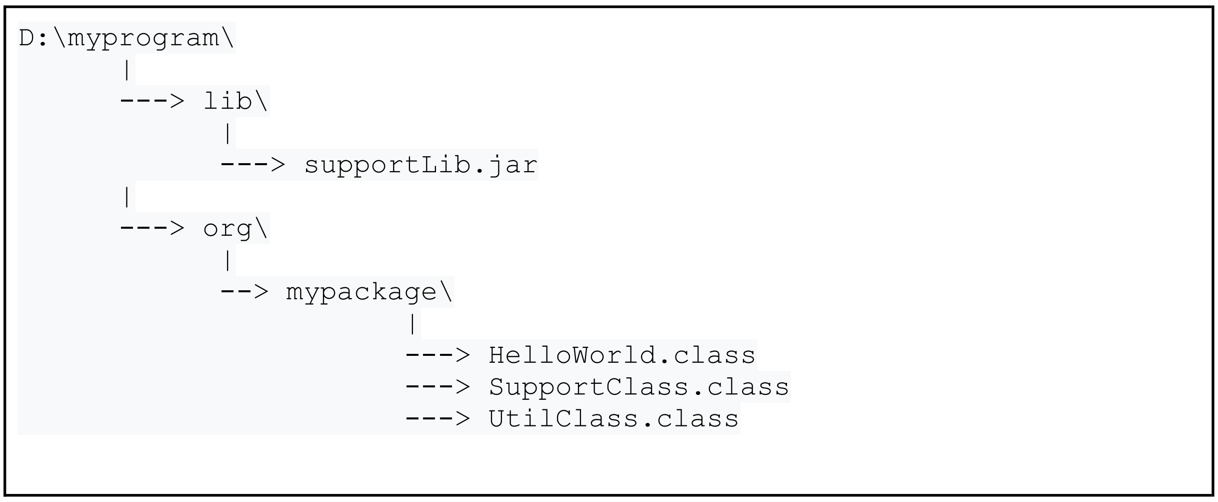
8.4 OpenJDK with classpath exception

OpenJDK was firstly developed and introduced in 2007 by the Oracle Corporation. Later, the Java community, IBM and Appl.Inc joined to work on its future versions. OpenJDK first edition was modeled on the basis of the simple JDK v 7.0, so it is the official , approved by the Free Software foundation and Open Source Initiative, implementation of the Java Standard Edition 7.

The OpenJDK with classpath exception is the Java implementation that was licensed under the GNU General Public License v 2.0 with a linking exception meaning that all components linked to the Java Class Library automatically fall under the GPL terms of conditions. The OpenJDK programmers work together on codes that are not only GPL-licensed but also approved by the Free Software Foundation and the Open Source Initiative.

Using OpenJDK with Classpath Exception appears to have more advantages in comparison to the other SDKs. Firstly, it is available for free online for every user. Secondly, one can improve the core library as he or she finds it appropriate for the personal benefits and purposes, automatically getting the GNU GPL license for the derivative Works. OpenJDK is more developer-oriented by releasing new features or updating them every six months

Along the information above, if the file is enclosed into the GNU GPL library, the typical location of the OpenJDK file with classpath exception has a physical appearance as it is illustrated on the image:



```
D:\myprogram\
       |
       ---> lib\
              |
              ---> supportLib.jar
       |
       ---> org\
              |
              --> mypackage\
                           |
                           ---> HelloWorld.class
                           ---> SupportClass.class
                           ---> UtilClass.class
```

*Image 4: "OpenJDK physical appearance" The image is taken from: **https://openjdk.java.net/groups/core-libs/***

The corresponding command for a computer in text format would be:

```
java -classpath D:\myprogram;D:\myprogram\lib\supportLib.jar
org.mypackage.HelloWorld
```

8.5 Advantages and Disadvantages of OpenJDK

The OpenJDK is in comparison with other software development kits - Oracle JDK or IOS Sbe DK, is a very flexible set of tools that is adjustable to the developers preferences and can be improved. Thus, any subsequent developer through the Java Virtual Machine optimization can achieve a faster performance, adopting it to the various software platforms such as FreeBSD, MacOS, Linux. The most important quality of the OpenJDK is the ability to be combined with other open-source software features and programs regardless of their original license. Additionally, improvements with the help of other library extensions on the top of the JDK license enhances the scalability of the OpenJDK (Sathyanarayanan, 2021).

Unfortunately, there are still some negative effects of the GNU GPL 3.0. The question at issue is that the OpenJDK, before using the library and distributing files under the Java label, the files with not original licenses has to be tested and successfully comply with the TCK -Technology Compatible Kit - conditions. Furthermore, one must accept the restrictions in the field of use by Oracle that leads to the conflict with the original GNU GPL, and may lead to some legal actions with the lost rights to execute the files under classpath linking (Riehle, 2011).

Second, if there is no need in the Java label, one can skip the TCK certification sacrificing the patent defense. That means that Java does not approve any patent rights contained in the software created with the OpenJDK code and a developer does not get a patent license without a valuable TCK aprovement. This way, a software developer exposes him or herself to the legal risks as an original patent holder can sue him or her for the redistribution of patents to the end users. The Image .. summarizes the problems which leads the absence of the TCK Certification (Riehle, 2011).

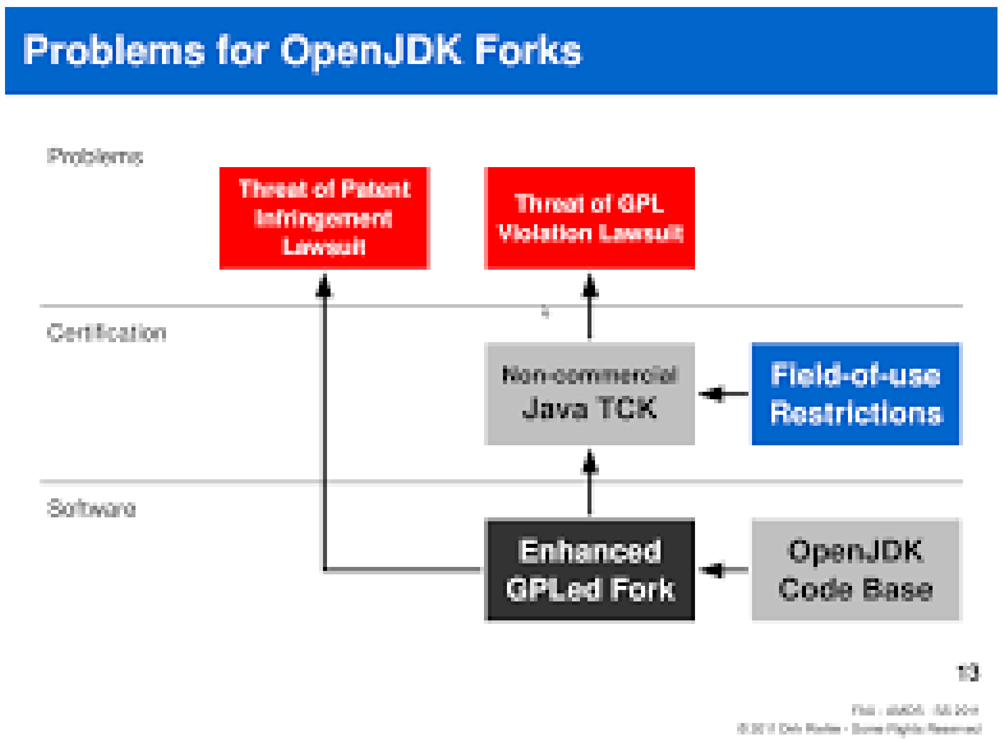

Problems for OpenJDK Forks
Problems
Threat of Patent Infringement Lawsuit
Threat of GPL Violation Lawsuit
Certification
Non-commercial Java TCK
Field-of-use Restrictions
Software
Enhanced GPLed Fork
OpenJDK Code Base
13

Image 5: "Problems for OpenJDK Forks" The image is taken from Riehle D. 2011. The Java IP Story.

9. Conclusion

To conclude, the entirety of the software foundation is grounded on the concepts of free-software and open source and the confusion that binds them. Whether one person is in pursuit of ethical movement or commercial gain, they will have to investigate the variety of licences and their limitations and conditions.

Despite everything, open-source software is currently gaining a greater popularity among the developers and end users due to its affordability, network effects, available open-source implementations, evolving environment and increase of developers' encouragement consequently result in higher adoptions rate worldwide. Open Source Software requires fewer costs and services then closed software licensing models. The open source applications and software are not dependent on the original creator or even the previous developer. Even if everything fails, the code will continue to exist and continue to be modified by users. Which is possibly due to approachable and convenient standards and requirements, therefore, it does not run into compatibility issues that could exist in proprietary counterparts. Developers are not constrained by any commercial restraints. Open-source programs are the product of collaboration which in turn will generate innovation for common advancement. On the other hand, open source licenses will be more vulnerable and susceptible to being hacked, than closed ones.

When it comes to the free software license, there is a big community surrounding it with mutual ethics and standards that are summarized in the 4 freedoms. A free license must always comply with the four freedoms to be considered a free program. Setting boundaries, eliminating some freedoms, and requiring users to pay is commensurate with exclusion and therefore acts as non-free.

Whether you choose an open source license like AL 2.0, free software like GNU, or a combination of both like CPL, you will have to refer to your own objectives and restrictions, which can be supplemented by the tables above (1-5).

9.1 The compatibility chart

The compatibility chart for the software licenses illustrates how different softwares under various licenses can be legally distributed (Wikipedia, 20219). It is constructed on the basis of the license's requirements and conditions, how the source code could be used by the subsequent developers and end users: “compatibility is a characteristics of two or more licenses according to which the codes distributed under these license may be put together in order to create a bigger distributable software” by Philippe Laurent (Wikipedia, 2021).

On this chart from Image 6 there are only open-source licenses presented. For Example, as the Image 6 below shows, Apache 2.0 as a permissive license is downstream compatible with the GNU LGPL v3, but is not legally permitted to be combined with the earlier LGPL version. The same way is Apache 2.0 not able to combine with the GPL licenses. Similarly, the GPL v 3.0 is upstream compatible with the LGPL v3.0 and downstream compatible with the Alfero Gpl.

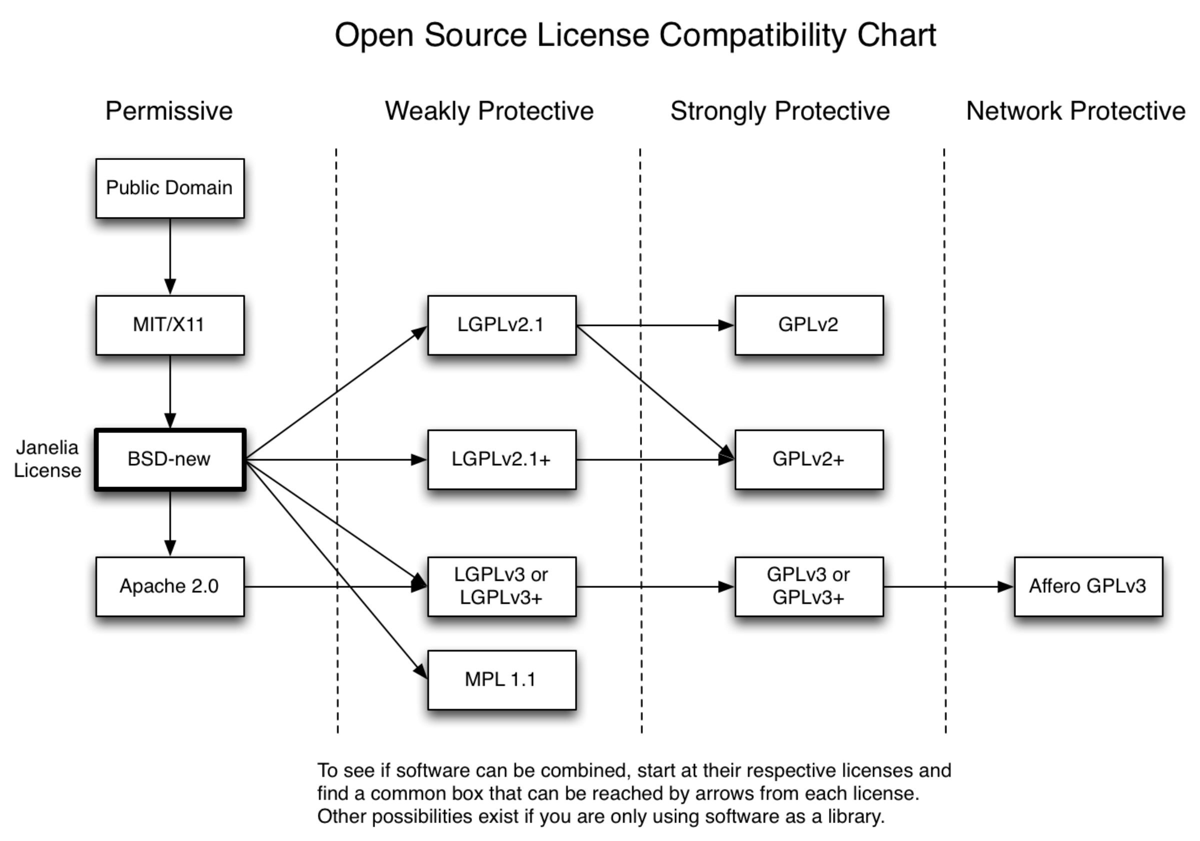

Open Source License Compatibility Chart
Permissive
Weakly Protective
Strongly Protective
Network Protective
Public Domain
MIT/X11
Janelia License
BSD-new
Apache 2.0
LGPLv2.1
LGPLv2.1+
LGPLv3 or LGPLv3+
MPL 1.1
GPLv2
GPLv2+
GPLv3 or GPLv3+
Affero GPLv3
To see if software can be combined, start at their respective licenses and find a common box that can be reached by arrows from each license.
Other possibilities exist if you are only using software as a library.

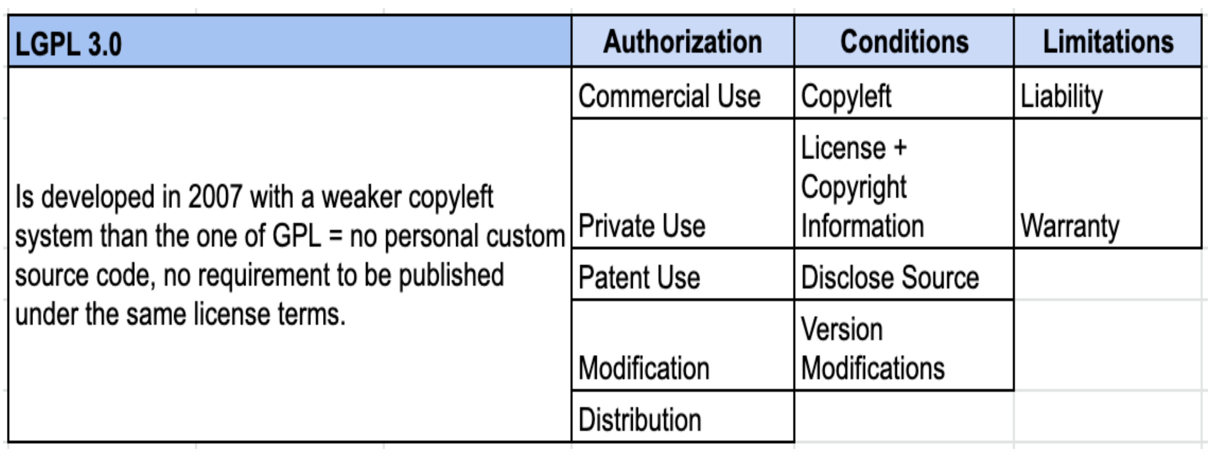
Image 6: "Open Source License Compatibility Chart" The image is taken from https://janelia-flyem.github.io/licenses.html

9.2 Overview of GPL 3.0, LGPL 3.0, AL 2.0, CPL 1.0 and OpenJDK with Classpath Exception

For a better overview of the summary of all covered licenses and comparisons, they are presented in a table form, where each license is compared with regards to the authorization, conditions, and limitations.

GPL 3.0 General Public License	**Authorization**	**Conditions**	**Limitations**
Is a free software license that employs 4 Freedoms philosophy. The fundamental facet is copyleft. Any software under GPL can be run for all purposes, commercial or not.	Distribution	Copyleft	No Warranty
	Modification	License Protection	Trademark Use
	Commercial Use	Disclose Source	Liability
	Private Use	Version Modifications	No proprietary programm integraion
	Patent Use		

Table 1: “GPL 3.0 General Public License”



LGPL 3.0	Authorization	Conditions	Limitations
Is developed in 2007 with a weaker copyleft system than the one of GPL = no personal custom source code, no requirement to be published under the same license terms.	Commercial Use	Copyleft	Liability
	Private Use	License + Copyright Information	Warranty
	Patent Use	Disclose Source	
	Modification	Version Modifications	
	Distribution		

Table 2: “LGPL 3.0”

CPL 1.0 - Common Public License	Authorization	Conditions	Limitations
Common Public License is a free software AND open-source software license published by IBM. The Free Software Foundation and Open Source Initiative both have approved the license terms of the CPL. Contributors may not remove or alter any copyright notices contained within the Program. Permission for distribution of a modified computer program is granted as long as the source code of a modified program are made available to others.	Commercial Use	Modification Notice	No Warranty
	Distribution	Copyright and Licensing notice + source code	No or conditional liability
	Modification		

Table 3: “CPL 1.0 - Common Public License”

APACHE LICENSE 2.0	Authorization	Conditions	Limitations
A permissive *open source* license that requires preservation and maintenance of its copyright and license information. It is widely adopted and refers to "Open Standards". Licensed works, larger works, and modifications are not under the same licensing terms and can be distributed in the absence of. Apache does not require companies to reveal technologies or skills used in order to construct a program on the basis of the work code	Commercial Use	Copyright and Licensing notice	Warranty
	Distribution		Trademark Use
	Modification	Version Modification	Liability
	Patented Use		
	Private Use		

Table 4: “Apache License 2.0”

OpenJDK with classpath Exception	Authorization	Conditions	Limitations
A free and open source application of the software development kit (a binary product constructed specifically for Java on Windows, Solaris, macOS and Linux). OpenJDK falls under GPL license and is approved by the Free Software Foundation and the Open Source Initiative.	Distribution	GNU/GPL conditions	TCK
	Modification		Oracle Certification

Table 5: "OpenJDK with classpath Exception"

10. Bibliography

Apache Foundation. Retrieved 20 March, 2021 from

https://www.apache.org/licenses/LICENSE-2.0

Burkett, D. (2016, June 12). *Software Licensing: Why It's Important and How It Can Help You.* Retrieved 29 March, 2021, from

https://workingmouse.com.au/innovation/software-licensing-why-its-important-and-how-it-can-help-you/#:~:text=Why%20Software%20Licensing%20Is%20Important&text=By%20acquiring%20too%20many%20software,manage%20software%20in%20your%20company

Crane, D. (June 13, 2019). *ActiveState. The Developer's Guide: Open Source Software License Comparison.* Retrieved 6 April, 2021, from

https://www.activestate.com/blog/the-developers-guide-open-source-software-license-comparison/

Eclipse Foundation. Retrieved 20 April,2021, from

Eclipse Public License 1.0 (EPL) Frequently Asked Questions | The Eclipse Foundation

Genesis Business Systems. The Risks of Using Unlicensed Software. Retrieved 20 April, 2021 from

https://www.genesisit.co.uk/blog/the-risks-of-using-unlicensed-software/#:~:text=Unlicensed%20software%20can%20carry%20viruses,of%20malware%20that%20infect%20computers.&text=Even%20if%20a%20piece%20of,through%20a%20lack%20of%20updates

Gilbert-Knight, A. (9March, 2012). Making Sense of Software Licensing. Retrieved 20 March, 2021 from

https://www.techsoup.org/support/articles-and-how-tos/making-sense-of-software-licensing

GNU General Public License. (2007). Retrieved 20 March, 2021 from
https://www.gnu.org/licenses/gpl-3.0.html

GNU General Puböic License, Version 2, With Classpath Exception. Retrieved 12 April, 2021, from

https://openjdk.java.net/legal/gplv2+ce.html

GNU Lesser General Public License, 20. Retrieved 10 April, 2021, from
https://www.gnu.org/licenses/lgpl-3.0.html

GNU Free Software Foundation. Why You Should Not Use The Lesser GPL for Your Next Library. Retrieved 2 May, 2021, from
https://www.gnu.org/licenses/why-not-lgpl.en.html

Free Software Foundation. (2021). *What Is Free Software?*. Retrieved 14 April, 2021 from
https://www.gnu.org/philosophy/free-sw.html.en

Free Software Foundation. (2021). *Warum “Open Source” das Ziele von Free Software verfehlt?*. Retrieved 14 April, 2021 from
https://www.gnu.org/philosophy/open-source-misses-the-point.html

Heather M. (5 January, 2020). *Open Source Software Licensing Basics for Corporate Users*. Retrieved 29 March, 2021, from

https://www.youtube.com/watch?v=gF4b1TA5Q5w

Jagielski, J. (12 November, 2015). Apache Software Foundation. Apache:Code, Community and Open Source. Retrieved 11 April, 2021, from

https://www.youtube.com/watch?v=KtGyfk1gK1A

Loosemore S, (n.d.) *The GNU C Library Reference Manual*. Retrieved 10 May, 2021, from

libc.pdf (gnu.org)

Milinkovic, M. (1 January, 2019). Eclipse Public License Version 2.0 Approved By OSI and Eclipse Foundation Board of Directors. Retrieved 16 April, 2021, from

Eclipse Public License 2.0 | The Eclipse Foundation

Open Source Initiative. Retrieved 29 March, 2021, from

https://opensource.org/licenses/cpl1.0.php#:~:text=THE%20ACCOMPANYING%20PROGRAM%20IS%20PROVIDED,RECIPIENT'S%20ACCEPTANCE%20OF%20THIS%20AGREEMENT

Oracle Corporation. 2021. Retrieved 2 May, 2021, from

https://openjdk.java.net/legal/gplv2+ce.html

Oracle Corporation, 2021. Retrived 06 May, 2021, from

Core Libraries Group (java.net)

Riehle, D. (2011). *The Java IP Story*. Friedrich-Alexander University of Erlangen Nürnberg. Retrieved 02 May, 2021, from

https://dirkriehle.com/wp-content/uploads/2011/06/The-Java-IP-Story.pdf

Wikipedia. (18n May, 2021). Retrieved 20 April, 2021 from

https://en.wikipedia.org/wiki/GNU_General_Public_License

Wikipedia. Retrieved 20 March, 2021 from

https://en.wikipedia.org/wiki/Comparison_of_free_and_open-source_software_licences

Wikipedia. (2021). Retrieved 20 April, 2021 from

https://en.wikipedia.org/wiki/Open-source_license#:~:text=An%20open%2Dsource%20license%20is,under%20defined%20terms%20and%20conditions.

Wikipedia. (2021): Retrieved on 9 April, 2021, from

https://en.wikipedia.org/wiki/Software_license

Winslow, S. (10 March, 2019). Dealing With Open Source Licenses. Retrieved 11 April, 2021 from

https://www.youtube.com/watch?v=KI5qBAWcQv8

Sathyanarayanan A. 2021. *EDUCBA. Oracle vs Open JDK*. Retrieved 30 April, 2021 from

https://www.educba.com/oracle-vs-openjdk/

Smith, B. (2014). A Quick Guide to GPLv3. Retrieved 24 March, 2021, from
https://www.gnu.org/licenses/quick-guide-gplv3.html#neutralizing-laws-that-prohibit-free-software-but-not-forbidding-drm

Tozzi, C. (15 June, 2016). *A Brief History of Free and Open Source Software Licensing*. Retrieved 21 April, 2021 from

Wallask, S. (September 2019). Source Code. Retrieved 16 April, 2021, from

https://searchapparchitecture.techtarget.com/definition/source-code

Wikipedia. (18n May, 2021). Retrieved 20 April, 2021 from

https://en.wikipedia.org/wiki/GNU_General_Public_License

Wikipedia. Retrieved 20 March, 2021 from

https://en.wikipedia.org/wiki/Comparison_of_free_and_open-source_software_licences

Wikipedia. (2021). Retrieved 20 April, 2021 from

https://en.wikipedia.org/wiki/Open-source_license#:~:text=An%20open%2Dsource%20license%20is,under%20defined%20terms%20and%20conditions.

Wikipedia. (2021): Retrieved on 9 April, 2021, from

https://en.wikipedia.org/wiki/Software_license

Wikipedia. 2021. Retrieved 15 May, 2021, from

https://en.m.wikipedia.org/wiki/License_compatibility

Wikipedia. (2021). Retrieved 14 April,2021, from

https://en.wikipedia.org/wiki/Free_Java_implementations

Winslow, S. (10 March, 2019). Dealing With Open Source Licenses. Retrieved 11 April, 2021 from

https://www.youtube.com/watch?v=KI5qBAWcQv8

Writer, S. (23 February, 2021). Sustainable Business Toolkit. *Software Licensing: Why It's Important and 3 Ways It Can Help You.* Retrieved 23 Match, 2021 from

Software Licensing: Why It's Important and 3 Ways It Can Help You