Contemporary Engineering Sciences, Vol. 7, 2014, no. 24, 1435 - 1443
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ces.2014.49176

Safe Architecture Design Strategies of the Proprietary Software Through Analysis of GPL License Family

Jun-Seok Park

Dept. of Computer Information & Communication Engineering
Sangmyung University, 20, Hongjimun 2-gil, Jongno-gu, Seoul, Korea

Soo-Hong Kim[*]

Dept. of Computer Software Engineering, Sangmyung University
31, Sangmyungdae-gil, Dongnam-gu, Cheonan, Chungnam, Korea

Abstract

Open Source Software(OSS) is being used extensively around the world with the emergence of its various advantages, but cases of violations of it are also increasing due to the lack of understanding in licenses. However, the researches on the violation of OSS licenses are focused on the areas of law, patent and economy, and it is rare to find a research paper on the violation of structural licenses that can occur in the actual development of software. We have analyzed cases of violation of licenses in GPL group, the typical OSS licenses, schematized it to develop a model based on which will be used to study architecture design strategies to avoid violation of copyright.

Keywords: Open Source License, Proprietary, Conflict, Violation, Snippet

[*] Corresponding Author

1 Introduction

Open Source Software (OSS) is freely available to anyone free of charge and is a program whose source codes are disclosed [1]. However it does not necessarily mean that borrowing the original copyright holder's source codes and reproducing them will be freely permitted. OSS's source codes are disclosed, but as shown in **Fig. 1**, they are works under the protection of intellectual property law such as copyright [2]. Nevertheless, many developers or users misunderstand and think they can use OSS licenses loosely since the source codes are disclosed free of charge providing little protection again infringement of copyright. Particularly, such violations are monitored world-wide by OSS organizations such as FSF (Free Software Foundation) [3], GPL-Violations.org [4] and SFLC (Software Freedom Law Center) [5] resulting in lawsuits for violations of various licenses. Companies such as Samsung Electronics and Humax was sued by SFLC in New York District Court [6] while LG CNS was sued directly by Steema Software from Spain [7].

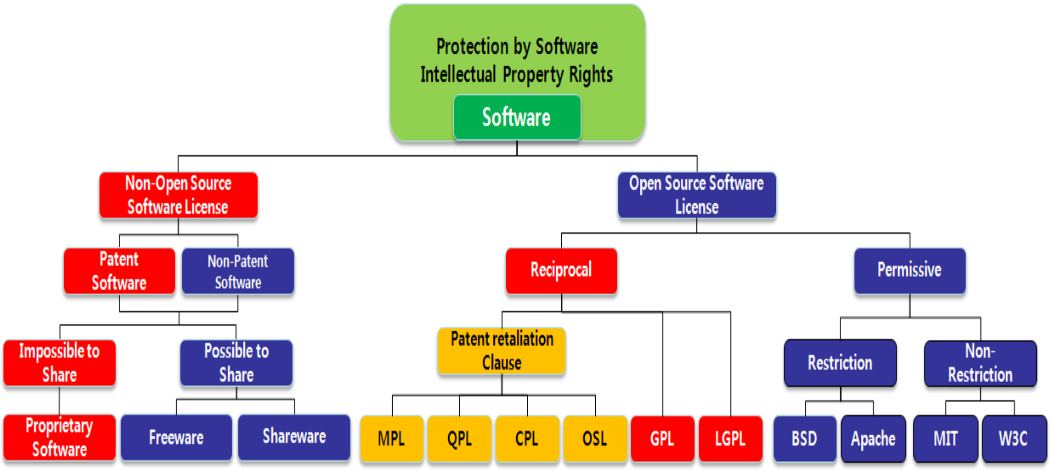

Fig. 1. Classification of Software

The researches in the field so far focused on stages prior to the development or after the distribution resulting in issues such as methods to classify licenses, considerations to apply licenses, the relationship between licenses and patents and legal solutions in cases of clashes between licenses rather than the point of development.

License issues and responses occurring due to snippets in files, module or partial codes made in the development of environment have rarely been the objects of research domestically or internationally.

One of the reasons would be the fact that the researches on OSS clashes have been conducted by legal and economics experts posing difficulties in studying the architecture from a perspective of software in-depth.

The research started with this point, analyzed violations of GPL-group licenses, the typical OSS licenses, applied the characteristics to build a schematized model, defined cases where licenses violation can arise or not and proposed appropriate design strategies based on them.

Chapter 2 reviews the flow of preceding researches on clashes between OSS licenses that were conducted so far, and chapter 3 analyzes requirements for GPL licenses. Chapter 4 schematizes models for each license using the analysis results of chapter 3 and proposes safe architecture design strategies of software applying them to arrive at conclusions.

2 Preceding Research

The existing researches on OSS mainly measured the economic impact of OSS or dealt with application methods of OSS. In particular, in the beginning of 2000s, a number of researches were conducted on the introduction of projects using OSS. Researches studying the economic impact of the open source movement and sharing by Lerner and Tirole are prime examples [8]. Coughlan studied development methods of sustainable community composition and cooperation in terms of business model from a perspective of project management methods [9]. However, this research applied general project management method to the OSS environment not escaping from researches on software development methods. Full-scale researches on the clashes between OSS licenses were conducted by Gordon who studied compatibility of OSS licenses. The research analyzed characteristics of difference licenses for each module existing within software module developed internally and covered issues of license compatibility and clash issues [10]. However, this research was limited to licenses regarding borrowed module falling short of analyzing characteristics of other projects. Technical researches of module units include a research of Mathur analyzing 1,423 projects for the possibility of clashes between source codes suggesting that GPL-group licenses occupy a high percentage of overall projects and fairly in accordance in a relative sense [11]. Nevertheless, the compliance of GPL licenses is too extensive to be decided by certain investigations for its rate of compliance lacking legitimate investigative methods. Also, the research studied possibility of clashes based on text matching in unit files or mathematical inferences failing to propose specific strategies to prevent clashes between licenses.

3 Analysis of the GPL License Family

No	Open Source Software License	Usage
1	GNU General Public License 2.0	27%
2	MIT License	16%
3	Apache License 2.0	15%
4	GNU General Public License 3.0	11%
5	BSD License 2.0(3-clause, New or Revised)	7%
6	Artistic License(Perl)	5%
7	GNU Lessor General Public License 2.1	5%
8	GNU Lessor General Public License 3.0	2%
9	Microsoft Public License(MS-PL)	2%
10	Eclipse Public License(EPL)	2%

Table 1. Open Source Software License Ranking [12]

The licenses that are mostly widely used among current OSS licenses are GPL-group licenses with a share of 45% as shown in **Table 1**. This is why this research selected GPL-group licenses as the model. When using OSS for proprietary software, a deep understanding of license requirements is a must. This chapter examines the characteristics of GPL-group licenses, the typical OSS licenses and utilized them as basic data to develop a model of disclosure scope of source codes.

GPL (General Public License) 2.0 is selected by the highest number of OSS presently, issued by FSF foundation, and awarded liberties to freely use, duplicate, distribute and modify. However, they must disclose all codes related to GPL codes, thus carrying more strict requirements than other licenses.

GPL (General Public License) 3.0 succeeded GPL 2.0 in essence, but added features related to DRM, software patents and compatibility for more flexible utilization. Notably, it was modified to be compatible with Apache License 2.0 that is used in various fields such as cloud computing, big data and Android.

LGPL (Lessor General Public License) is a strategically developed license to encourage OSS use through free use of OSS library. Basically, all codes related to LGPL codes must be disclosed as a LGPL license, but are not required to do so when they are linked by a library.

4 Architecture Design of the Proprietary Software using OSS

4.1. Model of disclosure scope of GPL and LGPL source codes

In the use of GPL source codes when developing proprietary software, source codes should be disclosed for all the program operated by a process in cases of modification of source codes or link of a new software. This involves inclusion to identical execution files and execution by a link to the shared address domain. LGPL is the same as GPL except that there is no obligation to disclose source codes when codes are linked. However, GPL loses a requirement to disclose source codes when two programs are communicated in the form of independent processes such as pipes, sockets and command-line arguments regardless of link. Certainly, deciding on the requirement to disclose cannot be made by any other than judges, but generally these are the exceptions that apply [13]. **Fig. 2** is a model of disclosure scope of source codes schematizing requirements of GPL-group licenses.

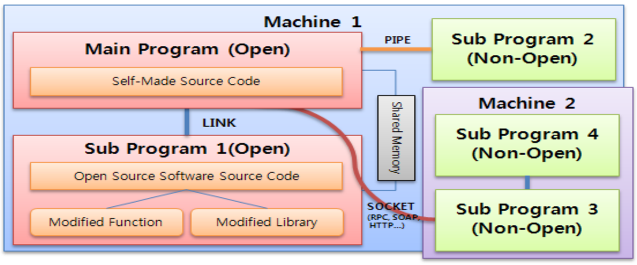

Fig. 2. Model of disclosure scope of GPL and LGPL source code

4.2 Architecture Design strategies of the Proprietary Software using OSS

Based on the above model, codes that can be disclosed in proprietary software include internally developed codes in the main program. In cases of using Linux system call, class path exceptions or loadable device drivers written in kernel module, there is no obligation to disclose. Additionally, when processes are written in independent forms using pipes, sockets and command-line argument methods regardless of link, the obligation to disclose sources codes expires even if they are linked to GPL.

Naturally, thorough consideration of content and meaning of communication will reveal that exchanging intricate internal data structures can be regarded as a large one program with two parts possibly producing exceptions. Nonetheless, most of the cases do not fall into the category of exceptions. **Fig. 3** is a model of disclosure scope in the case of link to GPL license.

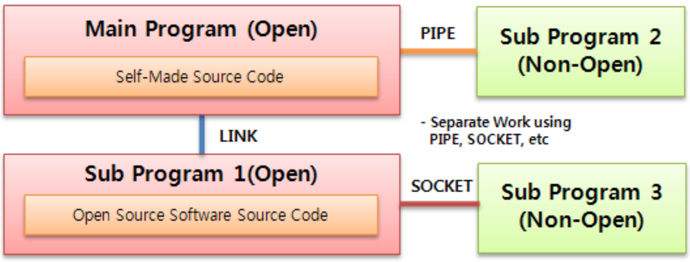

Fig. 3. Model of disclosure scope of GPL source code

To apply LGPL to proprietary software, codes that will be disclosed can be included in the internally developed codes in the main program while source codes that will not be disclosed can be maintained as link. However, with regard to LGPL, static link should provide object codes of application programs for user to modify library and run identical execution files. In other words, for the static link of sub program 1, object codes of sub program 1 should be disclosed. **Fig. 4** is a model of disclosure scope in the case of link to LGPL license.

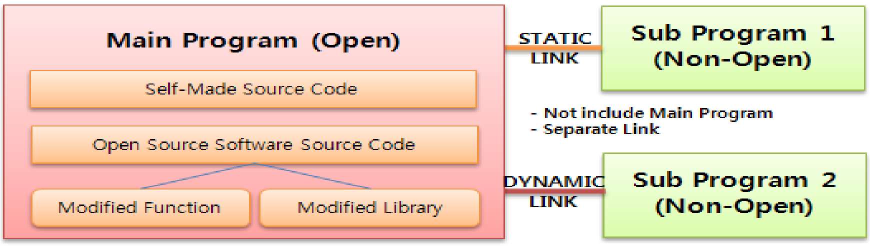

Fig. 4. Model of disclosure scope of LGPL source code

4.3 Architecture Design of Software linking GPL

This paragraph applies architecture design methods for each license discussed above and designs proprietary software linking GPL software. This case used Kernel Based Virtual Machine (KVM) to modify its execution software to link to internally developed software. In principle, the GPL sources codes that were modified and linked should be disclosed, but an exception applies since the execution methods were designed by calling socket communication and Linux system call. **Fig. 5** demonstrates the architecture structure of the software. The binary summoned by system call are designed as a separate process even in the shared memory domain, and therefore the internally developed source codes do not have to be disclosed.

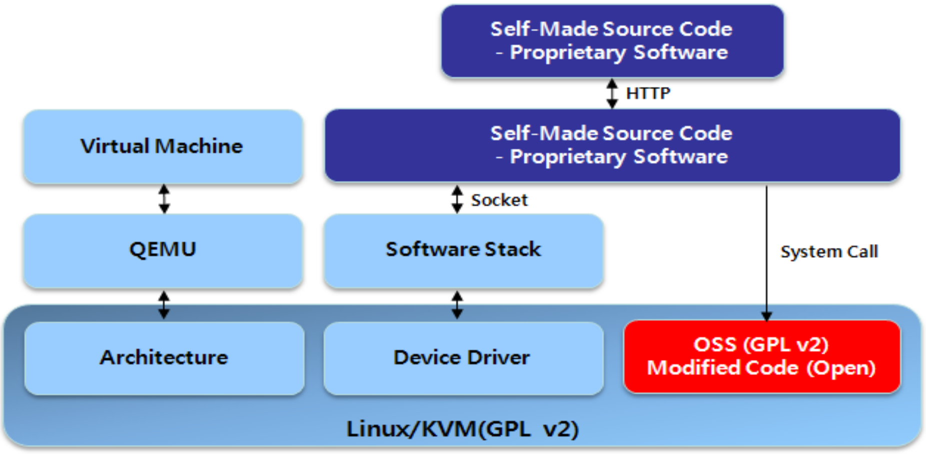

Fig. 5. Architecture Design method of the Proprietary Software using GPL

4.4 Wrong Architecture Design

Fig. 6 shows an example of architecture designed wrongly. The software included GPL library (A OSS Connecting Library) in the proprietary software, and must disclose all linked source codes in principle. Therefore, a separate library management module was created that is solely in charge of all tasks linked to GPL linkage library so as to not directly link main software and GPL codes evading a disclosure requirement of source codes of main software. It does not seem problematic at a glance, but this structure can result in an impression that main software is using library management module by link to include relevant module inside main software for distribution. That is, main software links directly to connecting library that are GPL source codes in the form resulting in the requirement to disclose all source codes according to GPL license.

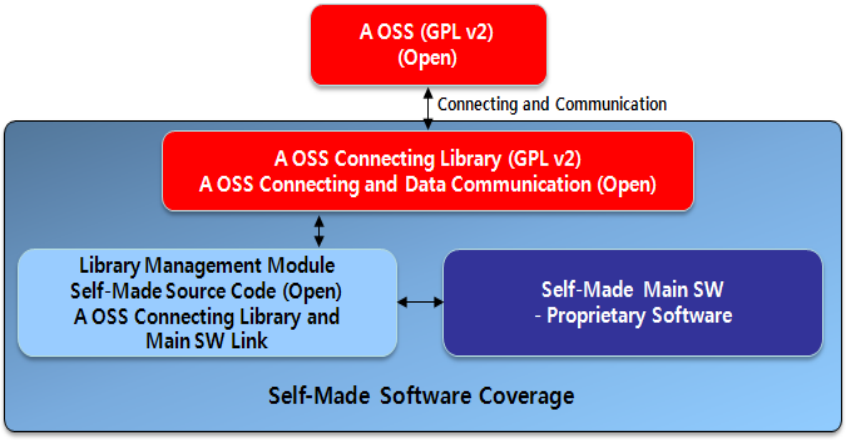
A OSS (GPL v2)
(Open)
Connecting and Communication
A OSS Connecting Library (GPL v2)
A OSS Connecting and Data Communication (Open)
Library Management Module
Self-Made Source Code (Open)
A OSS Connecting Library and
Main SW Link
Self-Made Main SW
- Proprietary Software
Self-Made Software Coverage

Fig. 6. Wrong Architecture Design

5 Conclusion

This research created a model schematizing requirements of GPL-group licenses to propose architecture design strategies for proprietary software utilizing the model. As can be seen in the architecture design wrongly linking GPL codes, architecture seeming appropriate at a glance can be a violation in reality, so through and careful examination of architecture when including OSS is necessary. Additional analysis of MPL, EPL, BSD and Apache License, not only GPL and LGPL, and the analysis of actual cases of violations through real software verification will lead to further researches on the architecture design strategies for general proprietary software utilizing OSS.

Acknowledgements. This work supported by the 2014 Sangmyung University research grant.

References

[1] Open Source Initiative, http://opensource.org/osd

[2] Andrew M., *Understanding Open Source & Free Software Licensing*, O’Reilly Publishing, 2004.

[3] Free Software Foundation, http://www.fsf.org

[4] GPL-Violations.org, http://gpl-violations.org

[5] Software Freedom Law Center, http://softwarefreedom.org

[6] Nancy Cogring, *Lawsuit Charges Open-Source License Violation*, IDG News Service, 2007.

[7] KimTong-hyung, *LG CNS Accused of SW Piracy*, The Korea Times, 2008.

[8] Lerner J., Tirole J., The Economics of Technology Sharing: Open Source and Beyond, *Journal of Economic Perspectives*, vol. 19 number 2(2005), pp. 99 – 120.

[9] Couglan, A Case Study of the Collaborative Approaches to Sustain Open Source Business Models, *Proceedings of the 9th International Symposium on Open Collaboration*, Article No. 17 (2013)

[10] T.F. Gordon, Analyzing open source license compatibility issues with Cameades, *ICAIL '11 Proceedings of the 13th International Conference on Artificial Intelligence and Law* (2011), pp. 51 – 55.

[11] Mathur A., An Empirical Study of License Violations in Open Source Projects, Software Engineering Workshop(SEW), *35th Annual IEEE*(2012), pp. 168 – 176.

[12] Blackduck Software, Top 20 Open Source License,
http://www.blackducksoftware.com/resources/data/top-20-open-source-licenses

[13] GNU Operating System, GPL License FAQ,
http://www.gnu.org/licenses/gpl-faq.html

[14] J. S. Park, S. H. Kim, A Study on the Safe Architecture Design Strategies of the Proprietary Software through Analysis of GPL License Family, *Workshop on Software 2014 Fourth*(2014), Jeju, Korea

Received: August 15, 2014