An investigation into the impact of software licenses on copy-and-paste reuse among OSS projects

Yu Kashima*, Yasuhiro Hayase†, Norihiro Yoshida‡, Yuki Manabe*, Katsuro Inoue*
* Graduate School of Information Science and Technology, Osaka University
Email: {y-kasima,y-manabe,inoue}@ist.osaka-u.ac.jp
† Deptartment of Computer Science, Graduate School of SIE, University of Tsukuba
Email: hayase@cs.tsukuba.ac.jp
‡ Graduate School of Information Science, Nara Institute of Science and Technology
Email: yoshida@is.naist.jp

***Abstract*—Because licensing an open source software (OSS) product restricts its reuse, the developer of the product has to consider the impact on reuse when choosing the license. However, to the best of our knowledge, there are no quantitative studies on the impact of software licenses on software reuse. To identify the impact, this paper presents a quantitative investigation into the relationship between the software license and copy-and-paste reuse on actual OSS products. The results show that the license of a product affects the frequency of reuse. On the other hand, copy-and-paste reuse occurs mostly in the source files distributed under the same license.**

***Keywords*-Software License; Open Source Software; Reuse; Copy and Paste**

I. Introduction

The source code of open source software (OSS) is available to anyone to modify or redistribute. Considering the growth in OSS development [1], software developers today have a huge amount of OSS source code available for reuse. Copy and paste (CnP) of code snippets is recognized as a simple and most casual reuse method which is performed frequently.

For various intents of OSS developers, there are many open-source licenses[1], which affect the usage frequency and situation of OSS products. Software reuse is also affected by the licenses since reuse is just another form of use.

When reusing existing software, the restrictions of both the license of the product being reused and that of the product being developed must be adhere to. In case the two licenses are incompatible, both cannot be satisfied simultaneously. For example, Apache license 2.0 (*Apachev2*) products cannot be incorporated into GNU General Public License version 2 (*GPLv2*) products, since several requirements in *Apachev2* conflict with a clause in *GPLv2*.[2] Furthermore, even if the licenses do not conflict, an OSS product cannot be reused if the license of the product being developed cannot be changed. For example, *GPLv2* source code cannot be incorporated into a 3-clause BSD license (*BSD3*) product. In contrast, *BSD3* source code can be incorporated into a *GPLv2* product because the restrictions of *BSD3* are included in those of *GPLv2* (Fig. 1).

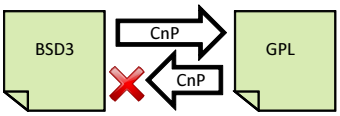

Fig. 1. Reusing source code in a different licensed product

Since switching the software license is complex and time consuming task, developers should pay attention also to the reuse frequency and situation when determining the license of a product. Especially, if an OSS project accepts the source code from many developers, it is very difficult to obtain consent of the all developers for the switching.

However, to the best of our knowledge, there is no quantitative study on CnP reuse from the point of view of software licenses. Developers currently determine the license without any underlying quantitative foundation. This paper presents the results of a large scale quantitative study on the relation between software licenses and CnP reuses based on the following two research questions.

- **RQ1** Is source code distributed under a permissive license reused more frequently than that distributed under a restrictive license?
- **RQ2** Which type of licensed source code more frequently imports source code of other OSS products?

The rest of this paper is organized as follows. Section II discusses the design and results of the two experiments, with an interpretation thereof. Finally, Section III presents our conclusions and future work.

II. Experiment

The goal of this study is to clarify the impact of software licenses on CnP reuse. In our previous study [2], we performed a preliminary investigation on the impact on a small Java source file set. To confirm the findings obtained from the previous study, this paper presents two experiments on large-scale file set: counting CnPs according to licenses, and statistical examination of the impact.

[1] the Open Source Initiative has officially approved 67 licenses. http://www.opensource.org/licenses (accessed Jan 2011)

[2] http://www.gnu.org/licenses/license-list.html (accessed Jan 2011)

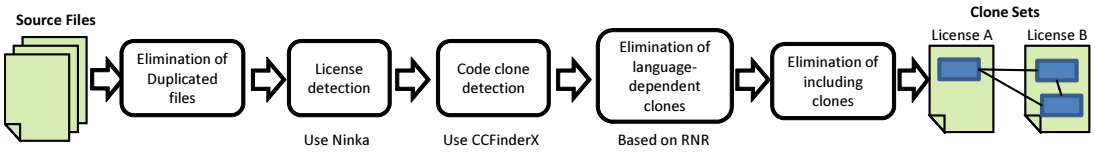

Fig. 2. Overview of the process for detecting licenses and CnPs

The outline of the two experiments is as follows. First, source files of OSS products are collected. Then, these source files are analyzed to detect instances of CnP. Finally, the instances are counted and assessed according to a certain criteria.

The following subsections describe design intent and procedure of detecting CnPs, analyzed code, and detail and result of the two experiments.

A. Detecting CnP – Design and Implementation

To clarify the impact of software licenses on CnP reuse, the license of the files and CnPs in the files must be detected. We designed the detection process as Fig. 2.

The detection process employs Ninka [3] for license detection and CCFinderX [4] for CnP detection. Ninka is employed since Ninka effectively and precisely detect the license automatically. CCFinderX is employed since CCFinder, the predecessor of CCFinderX, was used for CnP detection in previous studies[5][6]. Therefore, code clones detected by CCFinderX can be considered as code clones generated by CnP.

However, clones are not always suitable for counting CnPs, therefore, unsuitable code clones are removed from the counting. The clones which are not suitable for counting CnP is mostly classified into *language-dependent code clones* or *including code clones*. Examples of language-dependent code clones are consecutive if blocks, case entries of switch statements, and consecutive variable declarations. Language-dependent clones are usually generated not by CnP. With respect to the detail of language-dependent code clone, please refer to [7]. Including code clone is a code clone whose proper sub-fragment is a code clone. If including code clones are taken into counting, CnP is overcounted.

Detail of the five phases in the detection process (Fig. 2) is described below.

1) **Elimination of Duplicated files:** Eliminate identical files in target collection of source files except for one of them, because the experiment focuses on only copy-and-paste reuse of code fragments rather than reuse of entire files.
2) **License detection:** Identify the license of each file in the output of phase 1 using Ninka. The files that include no license description recognized by Ninka are eliminated and never passed to next phase.
3) **Code clone detection:** Identify the location of code clones using CCFinderX from the collection of source files delivered from phase 2.
4) **Elimination of language-dependent clones:** To eliminate language-dependent code clones, eliminate code clones which RNR (Ratio of Non-Repeated tokens) metric [7] is higher than 0.5 in the code clones detected in phase 3.
5) **Elimination of including clones:** Eliminate overlapped code clones which are included in other clones from the input from phase 4.

B. Analyzed Code

Two set of source files are analyzed in the experiments.

The first data set (**DS1**) contains all the C/C++ files in packages of *main* section in Debian/GNU Linux 5.0.2 lenny [8]. DS1 contains 776,289 files (286MLOC) obtained from 6,472 packages. DS1 is composed of common and widely-used OSS products.

The second data set (**DS2**) contains C/C++ source files sampled from SourceForge.net [9]. The sampled files are contained in 1,070 packages selected randomly which are developed in C/C++ and whose subversion repository has 10 or more commits. Selecting only packages whose repository has 10 or more commit is to exclude packages which have been seldom developed. As a result, DS2 contains 425,830 files (121MLOC). DS2 is designed as being representative of all the OSS products in the world.

DS1 and DS2 contain 41 same packages. These same packages do not spoil independencies of the two data sets, since the 41 packages are only 0.6% of DS1 and 4% of DS2.

From the obtained source files, the clones and licenses are detected. A brief summary of detection is shown in the Table II. The product license is explained using abbreviations of the license name for simplicity. Table I shows the basic abbreviations of the main licenses. “v” and the number immediately after a basic abbreviation denotes the license version. Additionally, a plus (+) sign immediately after the version number means “or any later”. If a product is distributed under a composite license such as dual-license or exception-clause, multiple license names are concatenated with commas and used as the license name of the product.

C. Experiment 1: counting clones for each license

Experiment 1 counts the number of code clones grouped by their license for both data sets. Through probing the differences between the counts, we try to reveal the relationship between CnP reuse and software licenses.

TABLE I
REPRESENTATIVE ABBREVIATIONS OF LICENSE NAMES

Abbreviation	Name
Apache	Apache Public License
BSD3	Original BSD minus advertisement clause
GPL	General Public License
LesserGPL	Lesser General Public License
LibraryGPL	Library General Public License
MX11	MIT License/X11License
MPL	Mozilla Public License
subversion	Subversion License

TABLE II
TOP 15 FREQUENT LICENSES

(a) DS1

	License	#Files
✓	GPLv2+	178,174
	LibraryGPLv2+	28,000
	LesserGPLv2.1+	24,540
	GPLv2	22,840
	GPLv3+	18,372
	GPLv2or LGPLv2.1, MPLv1_1	15,897
✓	BSD3	11,933
✓	MX11	11,715
	LesserGPLv2+	10,537
	boostV1	9,275
	GPLnoVersion	5,354
✓	Apachev2	4,297
	LibraryGPLv2	4,187
	BSD2	4,123
	LesserGPLv2.1	3,709

(b) DS2

	License	#Files
✓	GPLv2+	44,558
	boostV1	13,461
	GPLv3+	12,037
	LesserGPLv2.1+	8,765
	LibraryGPLv2+	6,705
	GPLv2	6,674
✓	Apachev2	6,220
✓	BSD3	4,784
	LesserGPLv2+	3,543
	LesserGPLv2.1	2,943
✓	MX11	2,478
	BSD2	2,408
	LesserGPLv3+	1,227
	FreeType	961
	subversion+	868

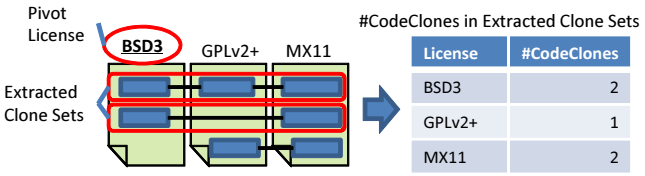

Fig. 3. Example of the Counting Code Clones in Experiment 1

TABLE III
#CLONES WITH PIVOT LICENSE IN DS1

(a) with Apachev2

License	#Clones
GPLv2+	38,520
Apachev2	15,618
GPLv2	5,261
LibraryGPLv2+	5,040
LesserGPLv2.1+	4,491
Others	25,814

(b) with BSD3

License	#Clones
GPLv2+	94,301
BSD3	66,271
GPLv2	13,066
LibraryGPLv2+	12,602
LesserGPLv2.1+	12,269
Others	71,851

(c) with GPLv2+

License	#Clones
GPLv2+	794,569
GPLv3+	114,476
LibraryGPLv2+	70,804
LesserGPLv2.1+	56,802
GPLv2	50,658
Others	245,805

(d) with MX11

License	#Clones
GPLv2+	104,542
MX11	84,482
LibraryGPLv2+	13,830
GPLv2orLGPLv2.1, MPLv1_1	13,738
GPLv2	13,639
Others	79,950

1) Method:

The following procedure is iterated until there are no further interesting licenses.

1) Manually select an arbitrary interesting license (the **pivot license**).
2) Extract the clone sets which include the pivot license clone.
3) Count the code clones in the extracted clone sets grouped by license (the **peripheral license**).

Fig. 3 illustrates an example of this step. There are three files together with their licenses. Boxes in the files denote clones, and these clones are connected by lines to compose clone sets. Let us assume *BSD3* is the pivot license. Since the upper two clone sets include *BSD3* clones, these sets are extracted. Then, the clones in the sets are counted grouped by license.

2) Results:

Distinguishing licenses are selected as pivot licenses from top popular licenses. (In Table II, pivot licenses are marked at the right of the name). Except the same type of license, four representative licenses are selected based on its popularity. Ultimately, pivot licenses are *GPLv2+*, *MX11*, *BSD3* and *Apachev2*.

Tables III and IV show the counts for DS1 and DS2 with each pivot license. Each table shows peripheral licenses and number of clones.

In all the resulting counts, the files distributed under *GPLv2+* contain most of the clones. In the result of DS1, the pivot license is generally second highest except in the case of *GPLv2+*. Similarly, in the result of DS2, the pivot license and *GPLv2+* ranks at first and second in the all counting results.

Meanwhile, number of source files strongly affects number of clones. To exclude the affection, the number of counted clones is normalized by dividing by the number of pivot license files and peripheral license files. The normalized value means the expected number of the clones in a peripheral license file when there are only one pivot license file and one peripheral license file. Table V shows the normalized values. If the value is higher, the cell has deeper color.

Table V shows that the expected number of clones is highest when the peripheral license is same to the pivot license in the case of *Apachev2*, *MX11* and *BSD3*.

Table VI shows number of clones per one pivot license file. These numbers suggest the tendency to be reused of each pivot license. The case of DS1 indicates that *MX11*, *BSD3* or *Apachev2* files tend to be reused frequently compared to *GPLv2+* files. Similarly, the case of DS2 indicates that *MX11*, *BSD3* files tends to be reused frequently compared to *GPLv2+*.

D. Experiment 2: statistical examination of licenses

The result of experiment 1 indicates the relationships between licenses and frequency of CnP reuse. For more detailed investigation, we statistically examine the impact of the license on CnP count using the same data set.

1) Method:

This experiment confirms whether the licenses affect to the number of CnP reuse even if the other factors which affect

TABLE IV
#CLONES WITH PIVOT LICENSE IN DS2

(a) with Apachev2

License	#Clones
Apachev2	9,336
GPLv2+	5,903
GPLv2	1,590
GPLv3+	1,466
LesserGPLv2.1+	1,016
Others	4,793

(b) with BSD3

License	#Clones
BSD3	8,686
GPLv2+	7,394
GPLv3+	1,667
LibraryGPLv2+	1,479
GPLv2	1,457
Others	7,002

(c) with GPLv2+

License	#Clones
GPLv2+	71,569
GPLv3+	6,643
LesserGPLv2.1+	6,143
GPLv2	5,360
LibraryGPLv2+	4,489
Others	24,418

(d) with MX11

License	#Clones
GPLv2+	6,985
MX11	5,086
LesserGPLv2.1+	1,276
GPLv3+	1,136
LibraryGPLv2+	1,128
Others	5,764

TABLE V
NORMALIZED VALUES FOR #CLONES(DEEPER COLORED CELLS DEPICT HIGH VALUES)

(a) DS1

	Pivot License			
	Apachev2	BSD3	GPLv2+	MX11
Apachev2	8.46E-04	3.30E-05	8.27E-06	3.72E-05
BSD3	4.28E-05	4.65E-04	1.11E-05	5.13E-05
GPLv2+	5.03E-05	4.44E-05	2.50E-05	5.01E-05
MX11	5.91E-05	5.90E-05	1.42E-05	6.16E-04

(b) DS2

	Pivot License			
	Apachev2	BSD3	GPLv2+	MX11
Apachev2	2.41E-04	3.23E-05	1.06E-05	4.00E-05
BSD3	2.09E-05	3.80E-04	1.32E-05	3.88E-05
GPLv2+	2.13E-05	3.47E-05	3.60E-05	6.33E-05
MX11	1.64E-05	2.82E-05	1.49E-05	8.28E-04

the reusability are removed. For the confirmation, two type of regression models are compared from the perspective of the fitness; One type of the models is only based on the factors which previous studies propose as reusability metrics, and the another type is based on both of license information and the reusability factors.

The response variable of the all regression models is the number of clones which relate to a certain file and which in an outside of a product. The counting rule is explained using the example in Fig. 4. There is a clone set between three products. Assume the case counting the number of related

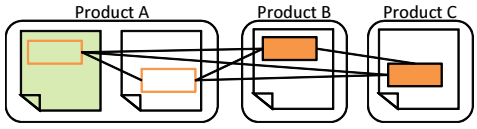

Fig. 4. Counting clones related to a certain file

TABLE VI
NUMBER OF CLONES FOR ONE SOURCE FILE DISTRIBUTED UNDER A PIVOT LICENSE

(a) in DS1

	#Clones	#Files	(#Clones) / (#Files)
Apachev2	94,744	4,297	22.04887
BSD3	270,360	11,933	22.6565
GPLv2+	1,333,114	178,174	7.482091
MX11	310,181	11,715	26.47725

(b) in DS2

	#Clones	#Files	(#Clones) / (#Files)
Apachev2	24,104	6,220	3.875241
BSD3	27,685	4,784	5.786998
GPLv2+	118,622	44,558	2.662193
MX11	21,375	2,478	8.625908

TABLE VII
THE METRICS FOR A FILE USED IN EXPERIMENT 2

Poulin's classification	Employed metrics
Complexity	LOC: lines of code excluding comment MCC: sum of McCabe's cyclomatic complexity
Documentation	COM: lines of comment
External dependencies	EXF: num of called functions defined in external EXV: num of used variables defined in external
Proven reliability	AGE: elapsed seconds since the file was created

clones focusing on leftmost file in product A. All clone sets that contains a clone in the focusing file are collected. The response variable is the number of clones in the collected clone sets excluding the product that have the focusing file. The clones in the same product are out of count because CnP in a same product rarely causes license problem.

Explanatory variables are licenses and the metrics which seems to relate to reusability. Table VII shows the 6 metrics employed as explanatory variables. The metrics are selected for covering Poulin's classification of reusability attributes [10]. Unfortunately, metric AGE is only applicable for DS2 since the repository of DS1 (Debian lenny) does not contain created time of the files.

Since license information is a nominal scale, license information must be transformed into *indicator variables*. Top 15 licenses shown in Table II are selected for the indicator variables for each data set.

Three regression models for each data set are made of abovementioned variables. The regression models are described below. Both the response variable and the metric values are logarithmically converted.

$$\log(\text{number of related clones} + 1) =$$

$$\epsilon + \sum_{m \in M} \alpha_i \log(m+1) \tag{M1}$$

$$\epsilon + \sum_{m \in M} \alpha_i \log(m+1) + \sum_{l \in L} \beta_i l \tag{M2}$$

$$\epsilon + \sum_{m \in M} \alpha_i \log(m+1) + \sum_{l \in L} \beta_i l + \sum_{m \in M} \sum_{l \in L} \gamma_{ij} l \log(m+1) \tag{M3}$$

TABLE VIII
ADJUSTED COEFFICIENT OF DETERMINATION VALUES

(a) for DS1

Model	R^2
DS1-M1	0.5021
DS1-M2	0.5047
DS1-M3	0.5133

(b) for DS2

Model	R^2
DS2-M1	0.3396
DS2-M2	0.3522
DS2-M3	0.3692

where M is a set of metrics, and L is set of indicator variables of Licenses.

The last clause of M3 means the interactions between the metrics and licenses. If the licenses affect CnP frequency, models M2 and M3 fit the data better than M1. The result of regression analysis is identified by the combination of data set and model names, e.g. DS2-M3 means the result of model M3 using dataset DS2.

As described above, the models are not straightforward linear expression but logarithmically converted. Straightforward linear models are discarded since they show far low fitness comparing to converted ones in our preliminary experiment. Generally, logarithmic conversion improves the fitness of regression models in many case of software repository analysis.

Finally, fitness of the model is compared. Difference between residual of the models are verified using ANOVA. If the significant differences are exist, *adjusted coefficients of determination* R^2 for the models are compared. If significant difference exists between two models, a model which has lager R^2 value fits to the data significantly better than another model.

2) Result:

First of all, differences between fitness of the models are tested by ANOVA. Tested pairs are <DS1-M1, DS1-M2>, <DS1-M2, DS1-M3>, <DS2-M1, DS2-M2>, and <DS2-M2, DS2-M3>. The comparison confirms that there are significant differences (p <2.2e-16) in the all pairs.

Since difference of the fitness is confirmed, the degree of fitness can be compared by coefficients of determination R^2. Table VIII shows that the R^2 value for each models. The R^2 values increase in order of M1 to M3 for both data sets. This result confirms that the license of a file has clear impact for CnP reuse even if the impacts of other factors are eliminated.

E. Revisiting Research Questions

1) RQ1: The experiment 1 show that *GPLv2+* code was distinctly less often reused than code with other license. Also, this result is supported by the significant impact of the license confirmed in the experiment 2. According to those results, we can conclude that *the source code distributed under a permissive license is more frequently reused than that distributed under a restrictive license*.

2) RQ2: In all 8 investigations on the two data sets in the experiment 1, files distributed under a particular pivot license are most often imported into files distributed under the same pivot license. Furthermore, Table V shows that the probability of CnP is highest when a pivot and target licenses are the same, except for *GPLv2+*. Therefore we can conclude that *source files that are distributed under a license are the most frequently imported into ones distributed under the same license*. On the other hand, *GPLv2+* was ranked as first or second in raw count, respectively. However, Table V does not support that *GPLv2+* files frequently import source code from other files. According to those results, we can conclude that *GPLv2+ files have a substantial impact on the reuse count because of available huge number of GPLv2+ files*.

III. CONCLUSION AND FUTURE WORK

This paper documents our study on the impact of software licenses on CnP reuse in C/C++ files. The results of the experiment show that CnP mostly occurs within source code distributed under the same license. On the other hand, substantial amount of reused code fragments are appeared in *GPLv2+* source code because the number of *GPLv2+* files is very large. Furthermore, the results confirms that non-copyleft files tends to be reused more than copyleft ones.

This paper focused on clone sets created by CnP. Hence, the direction of copying is not identified. For evaluating precisely the impact of the license on CnP, it is preferable for origin analysis to be performed to clarify the direction.

There are other methods for software reuse, such as library linking or file import. The relationship between software license and other reuse methods should be investigated, since these methods also cause license problems.

ACKNOWLEDGMENT

This work was supported by KAKENHI(21700031, 22800040, and 21240002).

REFERENCES

[1] W. Scacchi, "Free/open source software development: recent research results and emerging opportunities," in *Proc. of ESEC/FSE (Companion)*, 2007, pp. 459–468.
[2] Y. Kashima, Y. Hayase, N. Yoshida, Y. Manabe, and K. Inoue, "A preliminary study on impact of software licenses on copy-and-paste reuse," in *Proc. of IWESEP*, 2010, pp. 47–52.
[3] D. M. German, Y. Manabe, and K. Inoue, "A sentence-matching method for automatic license identification of source code files," in *Proc. of ASE*, 2010, pp. 437–446.
[4] T. Kamiya, "CCFinder Official Site," http://www.ccfinder.net/ccfinderx.html.
[5] D. M. German, M. D. Penta, Y.-G. Gueheneuc, and G. Antoniol, "Code siblings: Technical and legal implications of copying code between applications," in *Proc. of MSR*, 2009, pp. 81–90.
[6] H.-F. Chang and A. Mockus, "Evaluation of source code copy detection methods on freebsd," in *Proc. of MSR*, 2008, pp. 61–66.
[7] Y. Higo, T. Kamiya, S. Kusumoto, and K. Inoue, "Method and implementation for investigating code clones in a software system," *Information & Software Technology*, vol. 49, no. 9-10, pp. 985–998, 2007.
[8] Debian Project, "Debian GNU/Linux," http://www.debian.org/ Accessed Jan 2011.
[9] "SourceForge.net," http://sourceforge.net/.
[10] J. Poulin, "The search for a general reusability metric," in *Proc. of the Workshop on Reuse and the NASA Software Strategic Plan*, 1996.