

A Framework of Code Reuse in Open Source Software

Yuan-Hsin Tung, Chih-Ju Chuang, Hwai-Ling Shan

Telecommunication Lab., Chunghwa Telecom Co., Ltd., Taiwan, ROC

yhdong@cht.com.tw, wellmill@cht.com.tw, shanhl@cht.com.tw

Abstract—Recently, adopting open source software into software development has become a growing trend in the IT software industry. Unlike the general commercial software, open source software leverages the capability of the communities to improve the quality, but also reduce the cost of software development. This paper aims at analyzing the influence factors of applying open source software in software reuse. We begin by proposing our framework for enhancing the open source software usage and then set out to determine the influence factors of software reuse. A reuse system based on our proposed framework is implemented. For the assessment, a questionnaire survey and analysis of 20 software projects are carried out to evaluate our approach. There are 50 experienced corresponding project managers and developers participating in our experiments. The assessment shows that reuse factors of projects managers and developers influence the OSS reuse.

Keywords—open source software, software reuse, reuse factors, reuse model, reuse management

I. INTRODUCTION

Open source software (OSS)[25][38] is computer software with its source code made available and licensed, whereby the copyright holder provides the rights to study, change and distribute the software to anyone and for any purpose. In 1984, Richard Stallman proposed the GNU project[13], which was the first realization of open source software. With the support of multinational companies and governments, such as Google, Amazon, Oracle, Skype, LinkedIn, Facebook, NETFLIX, Open Source Software is becoming the mainstream of software development and maintenance to effectively cut costs, reduce development time, and increase software quality. According to the report of BlackDuck[3], the usage of open source software project increases from 250 thousand in 2010 to 850 thousand in 2012, and is over million in 2013. OSS reuse is a serious issue in enterprise software developing that should not be overlooked. A 2011 Gartner report [27] indicates that 50% companies will face the issues arising from OSS usage, which include OSS authority, licensing, appropriate source code to meet enterprise requirements, OSS quality, version control, and OSS management. IDC reports[15] in 2013 also presents the trend of OSS, and it observes that the creativity and innovation of open source software can be applied in mobile devices, cloud computing, and Big Data. All these point out that adopting open source software is indeed a growing trend in the IT software industry.

Software reuse has always been a major approach to increase productivity in software engineering, as it can provide several economic benefits, such as reducing software development time and operating costs, improving time-to-market and quality of software product, efficient utilization of development knowledge and corporate expertise[16][17][20].

Recently, to invigorate the technology community, adopting open source software is one of the recommended approaches for software reuse. Unlike the general commercial software, open source software leverages the capability of communities to improve the quality, and reduce the cost of software development. As more and more pieces of source codes are readily available[1] on the Internet, developers can easily access these resources and incorporate them as part of the software in development. Therefore how to properly adopt and reuse OSS becomes the challenge of software development[37]. One of the objectives of this paper is to identify the key influence factors in establishing an open source software reuse framework. Software reuse in OSS is not merely a technical issue, as there are many non-technical influence factors which need to be considered.

In this paper, we set out to determine the critical influence factors of software reuse in OSS so as to make reuse feasible. These influence factors will then be analyzed, and the associated difficulties of software reuse of open source software will be discussed. Based on earlier findings in the related works, we propose our OSS reuse framework and build accordingly our OSS reuse system to improve the software reuse processes. There are four main components in our framework, namely, software search, knowledge sharing, management, and human resource. A prototype system is constructed based on proposed framework. First, we define the expert forum to enhance the social discussion between developers. Then, the source code search engine mechanism is constructed. Finally, we examine source code of 20 projects and identify which open source licenses are used in these projects. In the assessment, via software projects examined and questionnaires, we determine the critical influence factors in corporate software reuse. Two experiments have been conducted to evaluate the performance of our approach. In experiment 1, we choose 20 projects and examine the source code of these projects. We identify the license agreements and open source software these projects used. The OSS usage statistics of the projects is collected to evaluate our projects. In experiment 2, based on the definitions of software reuse theory, we design our questionnaire according to BTOPP model. 50 experienced participants were involved in the experiment. The assessment shows that consideration of influence factors of project managers and developers can influence our proposed system.

The remainder of this paper is organized as follows: section 2 discusses related work. In section 3, a framework of OSS reuse is introduced to describe our proposed approach. Then we present our implementation of OSS reuse. In section 4, the assessment results of the questionnaire are explained. And we give our conclusion in Section 5.

II. LITERATURE REVIEWS

A. Open Source Software

The Open Source Initiative (OSI)[26], formed by Eric. S. Raymond and Bruce Perens in February 1998, is an organization dedicated to promoting the development and implementation of open source software. The Open Source Initiative defined the open source software as, "software that can be freely used, changed, and shared by anyone". Open Source Initiative (OSI) continued to present the "open source" cases to commercial businesses as they strived to bring major software businesses and other high-tech industries into open source community.

There are quite a few researches on the topic of open source software. In [22][34][31], the authors describe the key characteristics of OSS projects. OSS projects usually engage large numbers of volunteers. They can generally self-organize their work, and self-assignment is the most common mechanism for task management[4]. The requirement and development of OSS systems are usually quite informal [22][33], and the quality of OSS is attributed by the review process done by a large number of contributors as well as the expertise and passion of the developers on the work they choose to do [18]. In addition, the OSS license[26] defines usage of source code. This allows the end users to review and modify the source code for their own customization, curiosity or troubleshooting needs. GPL[12] (GNU General Public License) is the most prominent and popular OSS license. In license principle, open-source may have some restrictions, particularly regarding the expression of respect to the origin of software, such as a requirement to preserve the name of the authors and a copyright statement within the code, or a requirement to redistribute the licensed software only under the same license. Due to the associated potential risks, such requirements might sometimes prevent the enterprises from using open source software. And it is clear that the management of OSS license is essential in OSS reuse.

B. Software reuse model

In software engineering, software reuse[11][24][35] is considered a major factor for increasing productivity and quality. Software reuse may provide some economic benefits by reducing software development and operating costs, and by efficient utilization of development knowledge and corporate expertise[17]. In addition, software reuse can improve time-to-market, costs and quality of software products by ingraining reuse into the entire software development process. And the methodologies and approaches of software reuse are widely applied in enterprise. A large quantity of literature is dedicated to software reuse[6][7][17][21]. Reuse factors are factors that describe relevant aspects of software reuse, which can be related to both a technical and organizational issue. As shown in Fig.1, Morton [36] et al. proposed the reuse model, BTOPP, by organizing several reuse factors, Business, Technology, Organizations, Process and People.

- Business factors: The business factors are the indicator for the level of commonalities among software, such as "domain focus". Domain focus can influence the choice strategy of reuse component.

- Organization factors: Organization factors emphasize the elements that are identified at an organizational level, such as top management level supporting, organizational structure and reuse roles, and communication channels and organizational support.
- Process factors: The process factors can be directly related to a systematic reuse process, such as reuse planning, reuse supplier management, and quality management. In practice, additional mechanisms are usually used to enhance software reuse.
- People factors: There are many stakeholders in software development life cycle. The people factors should be considered in software reuse, such as developer skills and experience. In fact, the capability maturity of the developers is critical in software reuse.
- Technology factors: Technology factors focus on supporting tools for developers in software reuse, such as repository support, CASE tool support, and communication tools support. How to efficiently search and retrieve the appropriate solution is crucial in software reuse.

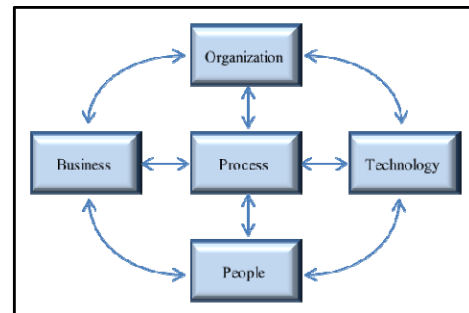


Fig. 1. BTOPP model [36]

C. Influence factors in OSS software reuse

There are many influence factors should be considered in OSS reuse. In [40], Georg von Krogh et al. analyze forms of knowledge reuse and the factors impacting on the open source projects. There are three forms of reuse open source code: 1) algorithms and methods; algorithms and methods are mathematical formula or a set of steps for solving a particular problem; 2) single lines of code; importing specific lines of code is a systematic and direct form of knowledge reuse in open source software projects; 3) components; the reuse of software component is usually provided by application programming interface (API), so that the component may exist autonomously from other components in a computer. The forms of code can influence the degree of willingness of the developers to reuse software.

In [40], the authors describe the efforts or cost to reuse open source code. To calculate the cost, three main stages in open source reuse processes have to be taken into consideration, namely, search stage, integration stage, and maintenance stage.

- Search stage: the objective in the search stage was to identify OSS software and components that would

improve the functionality and the performance of the software[8][19]. And the cost incurred by search must be compensated through the more effective innovation [39][40]. Developers search for appropriate software via developer communities, search engines[23], software repositories, and scientific databases. There are many OSS software search services provided on the Internet, such as Google Code Search[14], freecode (freshmeat)[9], and SourceForge[32].

- Integration stage[2] aims selecting and developing the dependencies among the reused components and systems. Different efforts of integration reflect in the choice of knowledge to be reused. The integration cost should be estimated after the search cost has been incurred and before the actual integration effort.
- Maintenance stage[40] follows integration in time and it also affects the choice of software reuse. In the enterprises, when developers reused the code in their system, they must frequently or continuously update bug fixes, improvements and new releases from the outside developers. The size of the community and the activity level informs the developer about the probability of future maintenance of the software they produce. Before a piece of software is reused, developer should sufficiently understand the software so as to decide on its integration and further improvements.

III. IMPLEMENTATION OF OSS REUSE FRAMEWORK

In this paper, to enhance the software reuse in open source software, we propose the framework of OSS reuse methodology. The research architecture is shown in Fig.2. We first survey the literature for analyzing influence factors of OSS reuse. Then, we propose our framework according to reuse model in section 3.1. Our framework contains four parts: software search, human resource, management, and knowledge sharing. We then construct our OSS reuse system according to framework by collecting COTS, open source software, enterprise principle, and prototype system in section 3.2. Finally, the experiments are conducted to evaluate our propose system.

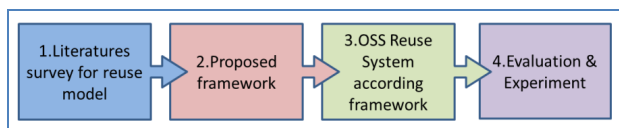


Fig. 2. Research Architecture of OSS reuse

A. OSS reuse framework

In this paper, we propose OSS reuse framework by referring to the BTOPP model and related work. We incorporate the influence factors into OSS reuse framework by analyzing the related work and considering requirements in practice. As shown in Fig.2, there are four main parts in our proposed framework: software search, human resource, management, and knowledge sharing.

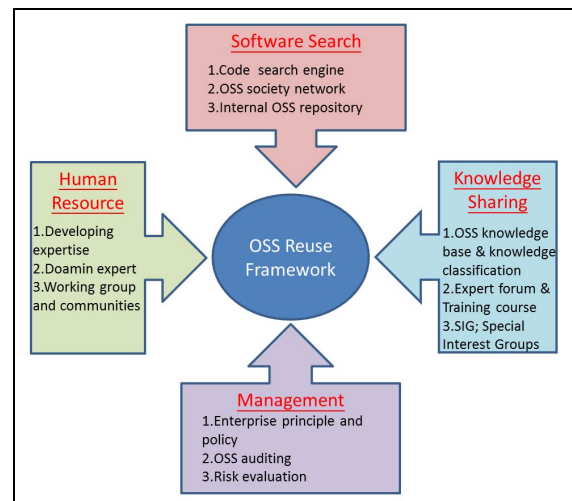


Fig. 3. Framework of OSS reuse

1) Software Search: Finding appropriate software is clearly quite important in software reuse. Basically there are two types of sources for software search mechanisms: external and internal. The external source covers assorted resources for finding appropriate software for specific purpose available on the Internet, such as search engines, web sites, mailing list, forum, academic journal, and expert committee. In fact, there are many famous search engines readily accessible to the developers, such as ohloh.net[28], google code search[14], freecode(freshmeat)[9], SourceForge[32]. With the internal source, we are referring to various facilities for enhancing software reuse with an enterprise, such as software database, e-learning system, software repository and library, and so on.

2) Human Resource: Human resource is an integral part for software development, as well as for software reuse. Training course and human resource management facilities are crucial for the success of software reuse. In addition to technical issues in OSS, there are also many non-technical issues in human resource which should be considered, such as training, coordination, collaboration, outsourcing, communities, working group, and so on.

3) Management: Management measurement contains many issues, such as license obligations, software auditing, vulnerabilities management, software repository, and management policies. Software auditing tools can help the developers better understand the code, such as BlackDuck Protex[3], FOSSology[10], Palamida[30], and OpenLogic[29]. With auditing tools, we can analyze the software for license obligations of open source software and identify well-known vulnerabilities and risks of software.

4) Knowledge Sharing: In the open source software development, there are usually a lot of participants working collaboratively. It is therefore important to provide a well-designed knowledge sharing mechanism. Common knowledge sharing channels include knowledge base[5], mailing list, forum, academic journal, and expert committee. In an

enterprise, a software repository is one of the most useful mechanisms to the developers for sharing their achievements.

B. System implementation for OSS reuse

In section 3.2, we construct our OSS reuse system based on the above framework by collecting commercial tools, open source software, enterprise principles, and prototype systems. And our system contains three main parts, OSS software search, OSS knowledge sharing, and auditing tool.

1) OSS software search: We adopt the Black Duck Customer Hub[3] as our resource search and OSS repository. The Customer Hub allows users to quickly and easily select OSS software and get OSS software information. Developers can quickly and easily get the answers that they are looking for. The information of Hub contains OSS component information, license obligations, vulnerabilities, community activities, update news, and software health. In our OSS repository, we propose 300 projects and components for developers and help developers get latest software easily.



Fig. 4. OSS search engine and repository; BlackDuck Customer Hub[3]

2) OSS knowledge sharing: In addition, we propose a knowledge sharing mechanism, such as knowledge bases, expert forum, OSS classifications. As shown in Fig. 5, we categorize the OSS software into 5 classes and 14 sub-classes. The 5 classes are application-layer, platform-layer, infrastructure-layer, framework, and IDE. And 14 sub-classes are contained in each class. For example, the platform-layer contains database, application, web-based, middle-ware, and so on. And we also categorize the OSS projects and components into various classes. We create forums for each class and provide the developers a space for discussion, as shown in Fig. 6.

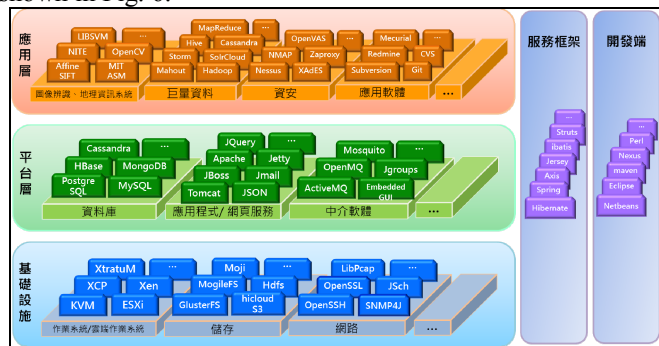


Fig. 5. Software architecture of OSS components

OSSElement	分類	數量	最後修改
資料庫(DB)	4	5	由 weathill 於 2013年 04月 30日, 16:39
應用程式(API)	2	3	由 weathill 於 2013年 04月 30日, 16:39
網頁應用(Web)	1	1	由 weathill 於 2013年 04月 30日, 16:06
中介軟體、被覆/軟體整合之OSS元件	5	6	由 weathill 於 2013年 04月 30日, 16:41
巨量資料(Big Data)	0	0	沒有文章
安全相關	0	0	沒有文章
影像/圖形、地理資訊系統(GIS)	0	0	沒有文章
網管、封包檢測、網路加速	0	0	沒有文章
雲端作業系統(Cloud OS)	0	0	沒有文章
儲存(Storage)	0	0	沒有文章
應用軟體	0	0	沒有文章
服務框架(Framework)	0	0	沒有文章
開發者相關	0	0	沒有文章
授權條款相關	1	1	由 weathill 於 2013年 04月 27日, 13:08
各位專家有無合適瞭解之 OSS，可先在此發表！	0	0	沒有文章
BlackDuck Protex: OSS輸入工具討論區	0	0	沒有文章
在 BlackDuck Protex 上提出問題	0	0	沒有文章

Fig. 6. knowledge sharing forums

3) Auditing tools: To manage OSS source code, we adopt the auditing tool, BlackDuck Protex[3], to scan our applications and check whether they contain any OSS source code. While the developers use open source software for free, it comes with license obligations that must be met. Auditing tool can help us to automatically check open source compliance of projects and reduce business risk of violating license agreement by analyzing software contents. In our system, we apply commercial software to analyze our own project. In addition, we can identify the vulnerabilities and potential risks of the software. In experiment 1, there are totally 20 projects scanned by the auditing tool.

IV. EXPERIMENT AND RESULT

A. Experiment Design

In this section, we design two experiments to evaluate the effectiveness of our proposed framework and implemented system. In experiment 1, we examine 20 software projects by BlackDuck OSS auditing tool, Protex. We try to determine the OSS reuse percentage of software projects in the period of 2013/01~2013/12. We can examine the participating projects which contain OSS software. In the experiment 2, we design the questionnaire and ask 50 experienced developers and project managers who are familiar with our implemented systems to take part in our experiment. The questionnaire is designed according to BTOPP model. There are 29 questions in five categories; Business, Technical, Organization, Process, and People. These participants (35 experienced developers and 15 project managers) are asked to decide which factors they think would influence software reuse. Ten-point Likert scale is used to evaluate the degree of satisfaction, ranging from strongly disagree (1) to strongly agree (10). The results from the questionnaire are shown below.

B. Result analysis

In experiment 1, there are 20 examined projects in six categories; communication, security, Quad-play, cloud computing, multimedia, multi-media, and intelligent community. As shown in Table 1, in our examined projects, there are 11 projects using over 40% OSS software in category C1, C2, and C3, which is the evidence that OSS is indeed widely used in software development. In addition, we analyze the top 30 OSS components of 20 projects. As shown in Table

2, the Web-based applications are most common reuse components and Apache license are widely used in our projects.

TABLE I. STATISTIC OF OSS USAGE IN 20 PROJECTS

Classification	Sum. of Proj.	Avg. Prog. Files	Avg. Prog. Size	OSS usage Percentage
C1	2	320	15MB	50%
C2	4	1124	268MB	52%
C3	5	657	112MB	44%
C4	3	1820	60MB	17%
C5	2	60	305KB	27%
C6	4	1733	142MB	22%
Sum.	20			

TABLE II. TOP 30 OSS COMPONENTS OF 20 PROJECTS

OSS Components	OSS Components	OSS Components
AJAX Control Toolkit	Apache Log4j	JBoss Transaction API
Apache Axis	CGLIB - Code Generation Library	jQuery
Apache Commons FileUpload	DOM4J - Flexible XML Framework for Java	JSON in Java
Apache Geronimo	Ext JS	JSTL - JavaServer Pages Standard Tag Library
Apache Jakarta Commons Codec	HBase	JTA - Java Transaction API
Apache Jakarta Commons Collections	Hibernate	JUnit
Apache Jakarta Commons IO	HTML Templates	MySQL Connector/J
Apache Jakarta Commons Logging	JAF - JavaBeans Activation Framework	OpenSSL
Apache Jakarta HTTP Client	JavaMail	Spring
Apache Jakarta Lang	JAXB 2.0 Project	Tomcat

In experiment 2, the scores of critical factors are shown in Table 3. The critical factors of highest average scores of are technical factors, 7.0 and 6.67 correspondingly. It indicates that the technical factor is considered as first priority by both project leaders and developers. At same time, the technical factors containing reliability, availability, release frequencies, and maintenance, are considered important factors by the participants. And the OSS search engine and OSS repository that we build can help developers to efficiently find appropriate components. In addition, the organization factors (average scores are 5.59 and 5.37) and business factors (average scores are 5.22 and 5.62) are ranked 2nd and 3rd factors to be considered in experiment 2, while people factors (average scores are 4.33 and 5.19) is ranked 4th in OSS reuse. In conclusion, to enhance OSS reuse, we need to include multiple factors, containing both technical and non-technical ones, to achieve OSS reuse benefits. Furthermore, the viewpoints of developers and project managers about critical factors are quite consistent.

TABLE III. RESULT ANALYSIS OF OSS QUESTIONNAIRE

No.	Question	PMs	Devel opers
1	The weight of OSS usage	5.1	4.9
2	Business factor		

2.1	OSS can reduce human resource cost in development	6.7	6.57
2.2	OSS can reduce commercial software license cost in development, including purchase, license, upgrade, and maintenance	6	6.29
2.3	OSS can shorten development time and improve development efficiency	7.3	8
2.4	OSS can reduce cost in system design	8	7.14
2.5	OSS can reduce cost in system integration	1.3	2.57
2.6	OSS can reduce cost in system maintenance	2	3.14
Average Score		5.22	5.62
3	Organization factor		
3.1	Support of high level management is important in OSS reuse	3.3	6
3.2	A task-specific organization is important in OSS reuse	7.33	4.85
3.3	OSS reuse management principle and policy is important in OSS reuse	5.33	4
3.4	Comprehensive OSS supporting tools is important in OSS reuse	4.67	4.85
3.5	A well technique sharing channel is important in OSS reuse	7.33	7.14
Average Score		5.59	5.37
4	Process factor		
4.1	OSS license will influence OSS reuse	8.67	9.14
4.2	A thorough reuse plan is important in OSS reuse	2.67	4.28
4.3	An auditing mechanism is important in OSS reuse	1	1.71
4.4	An OSS reuse repository is important in OSS reuse	4.67	3.71
4.5	An OSS component requirement system is necessary in OSS reuse	3.3	4.57
4.6	An OSS component quality monitoring is important in OSS reuse	4	5.14
Average Score		4.05	4.76
5	Technology factor		
5.1	An OSS components database is important in OSS reuse	4.67	6
5.2	Managing security risk is important in OSS reuse	8	6.57
5.3	The supports and activities of OSS community are important in OSS reuse	5.3	6.86
5.4	Reliability of OSS community is important in OSS reuse	8.67	7.71
5.5	The frequencies of OSS release is important in OSS reuse	8.67	7.43
5.6	A success case study is important in OSS reuse	6.67	5.43
Average Score		7.0	6.67
6	People factor		
6.1	The acceptance of new techniques by the developers is important in OSS reuse	4.67	5.71
6.2	Training materials of software are important in OSS reuse	3.3	6.28
6.3	Domain knowledge and maintenance capability of developers are important in OSS reuse	8	7.71
6.4	Expertise and experience of the developers is important in OSS reuse	6.67	5.71
6.5	An incentives for developers is necessary in OSS reuse	2	3.14
6.6	Personal achievement and expectation is important in OSS reuse	1.33	2.57
Average Score		4.33	5.19

V. CONCLUSION

Open source software is widely used in developing enterprise software, in which software reuse plays an important role. However, many influence factors should be

considered to ensure its success. In this paper, we proposed our framework to enhanced OSS reuse by surveying related reuse model. And we implement a reuse system according to this framework. In the experiment, we design the questionnaire and ask experienced developers and managers to participate. The evaluation result shows that reuse factors can influence our OSS reuse.

Open source software development has become a major trend of industry information systems development. However special attention to certain issues is required to maximize the benefits of software reuse while reducing the associated risk. In the future we will further focus on specific topic in the OSS software reuse, such as security related issue, vulnerabilities, and license obligations.

References

- [1] A. Mockus, R.T. Fielding, and J.D. Herbsleb, Two Case Studies of Open Source Software Development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*. 1(3) (2002) pp. 309-346.
- [2] Almeida, P., A. Phene, and R. Grant (2003) "Innovation and knowledge management: Scanning, sourcing, and integration", in M. Easterby-Smith and M. Lyles (eds.) *Handbook of organizational Learning and Knowledge Management*, Blackwell, Baskerville, pp. 356-371.
- [3] "Blackduck", <http://www.blackducksoftware.com/>, [Last accessed 1 May, 2014].
- [4] Crowston, K., Li, Q., Wei, K., Eseryel, U.Y. and Howison, J., "Selforganization of teams for free/libre open source software development," *Inf. Softw. Technol.*, vol. 49, pp. 564-575, 2007.
- [5] Donnellan, Brian, et al. "Implementing an open source knowledge base," *Software*, IEEE 22.6 (2005): 92-95.
- [6] D. Lucrecio, K. dos Santos Brito, A. Alvaro, et al. "Software reuse: the Brazilian industry scenario," *Journal of Systems & Software*. vol. 81, no. 6, 2008, pp. 996-1013.
- [7] Frakes, William, and Carol Terry. "Software reuse: metrics and models." *ACM Computing Surveys (CSUR)* 28.2 (1996): 415-435.
- [8] Fleming, L. (2001) "Recombinant uncertainty in technological search", *Management Science*, 47(1), pp. 117-132.
- [9] "Freecode", <http://freecode.com/>, [Last accessed 1 May, 2014].
- [10] "FOSSology", <http://www.fossology.org/projects/fossology>, [Last accessed 1 May, 2014].
- [11] Garlan, David, Robert Allen, and John Ockerbloom. "Architectural mismatch: Why reuse is still so hard." *Software*, IEEE 26.4 (2009): 66-69.
- [12] "GNU General Public License", http://en.wikipedia.org/wiki/GNU_General_Public_License, [Last accessed 1 May, 2014].
- [13] "GNU Project, Wikipedia", http://en.wikipedia.org/wiki/GNU_Project, [Last accessed 1 May, 2014].
- [14] "Google Code Search", <https://code.google.com/>, [Last accessed 1 May, 2014].
- [15] "IDC Research Reports", <http://www.marketresearch.com/IDC-v2477/>, [Last accessed 1 May, 2014].
- [16] Jacobson, I., Griss, M. and Johnsson, P. P. *Software Reuse, Architecture, Process, and Organization for Business Success*, Addison-Wesley, Reading, MA (1997).
- [17] Kotov, Vladimir. "Reuse In Software Development Organizations In Latvia." *Scientific Journal of Riga Technical University. Computer Sciences* 41.1 (2010): 90-96.
- [18] Lakhani, K. and Wolf, R.G., "Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects," *SSRN Electronic Journal*, 2003.
- [19] Majchrak, A., L.P. Cooper, and O.P. Neece (2004) "Knowledge reuse for innovation", *Management Science*, 50(2), pp. 174-188.
- [20] Morad, Shlomit, and Tsvi Kuflik. "Conventional and open source software reuse at Orbotech-an industrial experience." *Software-Science, Technology and Engineering*, 2005. Proceedings. IEEE International Conference on. IEEE, 2005.
- [21] Morton, M.S. *The Corporation of the 1990s: Information Technology and Organizational Transformation*. Oxford University Press, USA, 1991.
- [22] Mockus, A., Fielding, R.T. and Herbsleb, J., "A case study of open source software development: the Apache server," *ICSE '00: Proceedings of the 22nd international conference on Software engineering*, New York, NY, USA: ACM Press, p. 263—272, 2000.
- [23] McMillan, Collin, et al. "Exemplar: A source code search engine for finding highly relevant applications." (2012): 1-1.
- [24] Morisio, Maurizio, Michel Ezran, and Colin Tully. "Success and failure factors in software reuse." *Software Engineering, IEEE Transactions on* 28.4 (2002): 340-357.
- [25] "Open-source software, Wikipedia", http://en.wikipedia.org/wiki/Open-source_software, [Last accessed 1 May, 2014].
- [26] "The Open Source Initiative", <http://opensource.org/>, [Last accessed 1 May, 2014].
- [27] "Open Source-Gartner", <http://www.gartner.com/technology/core/products/research/topics/openSource.jsp>,
- [28] "ohloh.net", <http://www.ohloh.net/>, [Last accessed 1 May, 2014].
- [29] "OpenLogic, Open Source Scanning Solution", <http://www.openlogic.com/solutions/open-source-scanning/>, [Last accessed 1 May, 2014].
- [30] "Palamida", <http://www.palamida.com/>, [Last accessed 1 May, 2014].
- [31] Raymond, E.S., *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, O'Reilly & Associates, Inc., 2001.
- [32] "SourceForge", <http://sourceforge.net/>, [Last accessed 1 May, 2014].
- [33] Scacchi, W., "Understanding the requirements for developing open source software systems," *IEE Proceedings - Software*, vol. 149, pp. 24-39, 2002.
- [34] Sethanandha, Bhuricha Deen, Bart Massey, and William Jones. "Managing open source contributions for software project sustainability." *Technology Management for Global Economic Growth (PICMET), 2010 Proceedings of PICMET'10: IEEE, 2010*.
- [35] Spoelstra, Wouter, Maria Iacob, and Marten van Sinderen. "Software reuse in agile development organizations: a conceptual management tool." *Proceedings of the 2011 ACM Symposium on Applied Computing*. ACM, 2011.
- [36] Schwartz, R.B. and Russo, M.C. How to quickly find articles in the top IS journals. *Commun. ACM*, 47 (2). 98-101.
- [37] S. Hissam, C.B. Weinstock, D. Iakosh and J. Asundi, *Perspectives on Open Source Software*, technical report CMU/SEI-2001-TR-019 ESC-TR-2001-019 (2001).
- [38] St. Laurent, Andrew M. (2008). *Understanding Open Source and Free Software Licensing*. O'Reilly Media. p. 4.
- [39] von Krogh, G., Ichijo, K. and Nonaka I. (2000) *Enabling Knowledge Creation: How to Unlock the Mystery of Tacit Knowledge and Release the Power of Innovation*, Oxford University Press, New York and Oxford.
- [40] von Krogh, Georg, Sebastian Spaeth, and Stefan Haefliger. "Knowledge reuse in open source software: An exploratory study of 15 open source projects." *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on. IEEE, 2005*.