Reflection on Knowledge Sharing in F/OSS Projects[1]

Sulayman K. Sowe and Ioannis Stamelos
Department of Informatics, Aristotle University, Greece
sksowe,stamelos{@csd.auth.gr}

Abstract. Knowledge sharing between software project participants simplifies a range of decision-making processes and helps improve the way software is being developed, distributed, and supported. However, research in this area has traditionally been very difficult because the source of knowledge, the code, has been a guarded secret and software developers and users inhabit different worlds. F/OSS projects have changed the way we perceive and understand knowledge sharing in distributed software development. This short paper presents our current understanding, and what needs to be done in terms of empirical research in knowledge sharing in F/OSS projects.

KeyWords: Open Source Software Development, Open Source Software Projects, Knowledge Sharing, Knowledge Resources, Collaborative Software Development.

1 Introduction

Free and Open Source Software (F/OSS) has changed the way we initiate software projects, develop, distribute, and maintain software. In a bazaar styled project roles are not clearly defined and software users are treated as co-developers. The cohabitation of both developers and users facilitates communication between all individuals involved in the software development process, thus, making it easy to study the knowledge sharing activities. Many F/OSS projects succeed because people volunteer their time and effort to share their knowledge and develop the software. Interaction between participants in the form of ”talking to each other” is important for cooperation, resolution of conflicts, and the establishment of trust between project participants [1]. If F/OSS is all about coding and few project members are willing to interact and share their knowledge, questions one may ask are;

[1] This research is partly funded by the FLOSScom project (http://flosscom.net/). Grant No. 229405-CP-1-2006-1-PT-MINERVA-MPP

- effective communication and knowledge creation are used as measures of project success [2,3]. But is there knowledge sharing, who are the people involved, how much knowledge sharing are developers and users doing?
- how can participants at the very edge of the project's community feel that they are part of the project when they are not opportune to interact and share their knowledge with the rest?

The first step in answering these questions is to understand the nature and dynamics of knowledge sharing in F/OSS projects. The rest of the paper attempts to develop such understanding. First we argue for a paradigm shift from traditional knowledge sharing practices. Then we address knowledge sharing in F/OSS projects and use some empirical studies to reflect on our current understand and map future research directions.

2 Paradigm Shift in Knowledge Sharing in F/OSS

The importance of knowledge sharing is not unique to F/OSS projects alone. Software projects face a problem of how to leverage and transform the tacit knowledge of community members into explicit usable knowledge. Practically, a successful project needs to manage its knowledge resources (people and technologies) in order to generate an efficient and sustainable software development environment. Knowledge management (KM) practices often consider how organizational structures and processes promote collaborative learning [6] and sharing of knowledge [4]. The goal of knowledge management is to lower barriers imposed by organizational structures, format restrictions, and conceptual limitations, so that the acquisition, transfer, and sharing of knowledge can be facilitated. Thus, KM helps us understand tools, roles, and activities we need to develop in order to help individuals benefit from F/OSS. However, F/OSS projects are different entities when compared to organizational structures in which many KM studies are conduct. To address KM issues in F/OSS projects, one needs a *paradigm shift* from organizational attention and knowledge making as a manufacturing process to online interactive systems where knowledge making is characterized by the virtual presence of geographically distributed groups of volunteers. The virtual world makes hording and selling knowledge difficult, if not impossible, because someone else might be positioned to provide the same knowledge for free.

The F/OSS development process not only highlights a different way of sharing knowledge but also acquiring it is a complicated process and building on existing knowledge takes time. Software development is a human-based and knowledge-intensive activity [8,9]. In addition to sound technological infrastructure and effective tools to coordinate the collaborative software development process [1], the success of any project immensely depends on the knowledge and experience brought to it by software developers and users. F/OSS projects provide the means for members to transform their tacit knowledge into explicit [4] (**Fig. 1**). As illustrated, project participants socialize by sharing their knowledge. Individuals make their tacit

knowledge explicit to the project through externalization. Combination refers to the formation and organization of abstract knowledge from explicit knowledge. Through internalization individuals will absorb explicit knowledge, combining it with their own knowledge and experiences to produce "new" tacit knowledge. Through active participation, individuals' tacit knowledge is transformed into explicit knowledge.

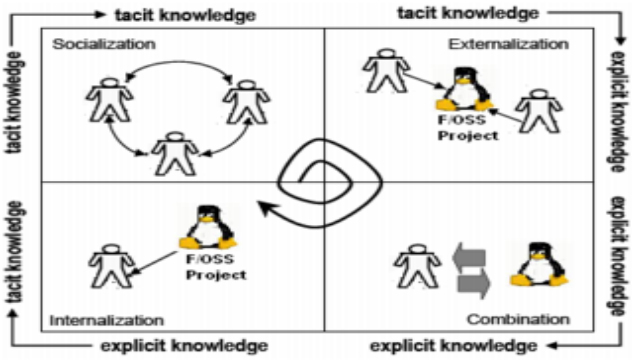


Fig. 1. Making tacit knowledge explicit in F/OSS projects

F/OSS projects are complex environments with many different actors. There is no indication that all project participants will interact and share their knowledge. When members do, *knowledge link* (***K***) is said to exist between them. The link is a reciprocal relationship of the form *A talks to B* and *B talks to A* and can be expressed as; $K_{AB} = \{^{1}_{0}$; *where* $\boldsymbol{K}_{AB}$*=1 if there is knowledge sharing between actor A and actor B, and 0 if otherwise.*

Sharing knowledge is a synergistic process. If a project participant shares his ideas or a way of configuring particular software with another person, then just the act of putting his idea into words will help him shape and improve his idea. If he enters into a dialogue with the community, then he may benefit from their knowledge, from their unique way of doing things and improve his ideas further. The benefit derived from knowledge sharing is that participants learn from each other, and the result of their interaction is archived in the project's repositories from which subsequent participants can learn.

3 Knowledge Sharing in F/OSS Projects

The F/OSS development process facilitates the creation, diffusion, and transformation of software knowledge at a rate unprecedented in the history of software development. The prospects for expert software developers and novice users to understand the software development process are now great. However, our understanding of knowledge sharing in F/OSS projects is challenged by the fact that individuals or even infrastructures are often located at large distances from each other, making tra-

ditional face-to-face knowledge sharing practices hard. Furthermore, the "babbling bazaar of differing agendas and approaches" [10] in some projects has people with varying knowledge and skills. Nevertheless, the F/OSS development process has made the empirical study of knowledge sharing possible because we can now use the freely available data in mailing lists, forums, code versioning systems, bug trackers, etc. to understand who is involved, who is talking to whom, and what is talked about. The F/OSS development process is knowledge intensive and involves real people who create, maintain and evolve the software. Therefore, focusing on knowledge sharing in F/OSS projects may shed light on pending research issues such as motivation and involvement of developers and users, community and software project formation and dynamics, reasons for projects success or failure, software engineering practicing, learning opportunities, how participants' contributions change overtime, etc.

For many individuals, developing new skills and sharing knowledge are primary motives for participating in F/OSS. Through reflective practice and collaborative learning, participants strive to improve the software in their respective projects, develop and refine their skills, challenge and question themselves. There are many opportunities for serendipity or information encountering in F/OSS projects. For example, a potential knowledge seeker [6] confronts an unfamiliar concept (or bug) in the use of an application (e.g. OpenOffice) and decides to seek help form a project's mailing list. There are two ways to look at this scenario. First, if the concept has been encountered before, it will be captured and stored in the project's knowledge base (mailing lists, forums, code repository, etc). He/she then consults and learns from the knowledge base. Second, if the concept has not been encountered, ie. the knowledge seeker's problem has not been addressed in the project before. He/she then identifies the appropriate list, posts his/her question, and exchanges ideas with list participants. Alternatively, the knowledge seeker may know a project participant with expertise in the area he is interested in and exchange direct emails with that person, with or without copies being posted to the list.

4 An Empirical Investigation of Knowledge Sharing

Methodologies suitable for investigating knowledge sharing in terms of email exchanges have been proposed [5, 6]. The intangible nature of tacit knowledge makes it very difficult, if not impossible, to measure. However, when tacit knowledge is transformed into explicit knowledge through socialization or interaction and shared by members of a project, an attempt can be made to measure how much knowledge is being shared. One approach to quantify the level of knowledge sharing in F/OSS is by analyzing substantial email exchanges between list participants. This can be achieved by

(i) Counting the total number of posts externalized to the list. That is, email messages potential knowledge seekers posted. We represented this value by the *nposts* variable.

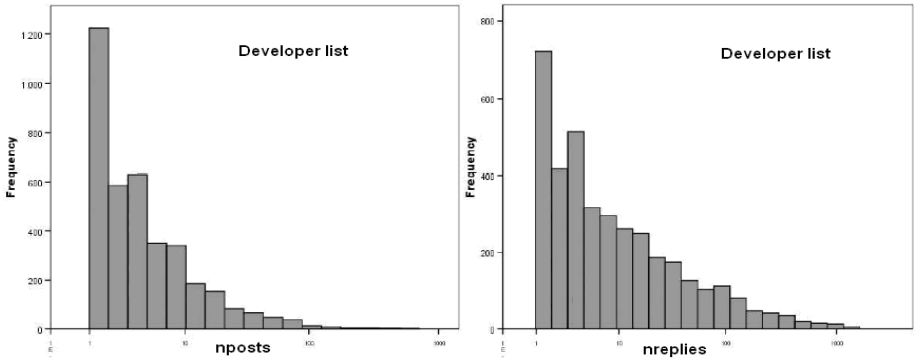
(ii) Counting the total number of replies made by potential knowledge providers to questions posted to the lists. This value is represented by the *nreplies* variable.

The data (**Table 1**) collection, extraction, and cleaning methodology for investigating knowledge sharing in mailing lists is obtained from the Developer and User mailing lists of the Debian project [6]. The dataset was collected over 5 years (January 2000-December 2005) and consists of 3735 participants in the Developer list, who posted 29685 email messages, which generated 128933 replies. The 5970 participants in the User list posted 193276 email messages, which generated 165380 replies.

Table 1. Descriptive statistics of Developer and User mailing lists data.

		Developer	User
nposts	Mean	7.95	32.37
	Median	3.00	7.00
	Std. Dev.	21.302	121.753
	Maximum	523	4106
nreplies	Mean	34.52	27.70
	Median	6.00	5.00
	Std. Dev.	105.567	122.040
	Maximum	1517	4168

Comparatively, the mean (posts/person) and median of *nposts* are smaller for the Developer list, while the same measures have larger values for *nreplies*. Thus, the posting and replying activities of the participants are different in the two lists - further clarified in **Fig. 2**.



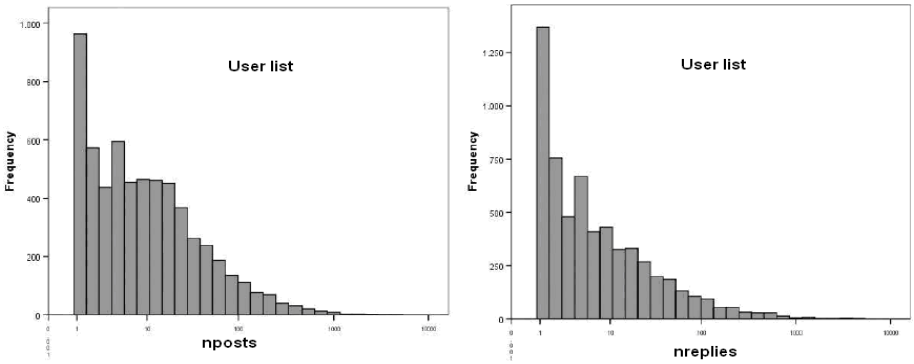


Fig. 2. Frequency distributions of posts and replies.

Participants in the User list contributed small posts and small replies. For the developer list, participants' activities were characterized by a small number of posts and a large number of replies. For the relationship between posting and replying activities, Table 2 shows only the Kendall's *taub* and Spearman's *rho*, since we ranked the participants according to their positing or replying [7].

Table 2: Nonparametric correlations between posts and replies in the two lists (Sig. =0.000 for all correlations).

			Developer List		User List	
Test	Variable		nposts	nreplies	nposts	nreplies
Kandella*'s taub*	nposts	Corr. Coef.	1,000	,475*	1,000	,550*
	nreplies	Corr. Coef.	,475*	1,000	,550*	1,000
Spearmans *rho*	nposts	Corr. Coef.	1,000	,608*	1,000	,699*
	nreplies	Corr. Coef.	,608*	1,000	,699*	1,000

5 Reflection on Knowledge sharing in F/OSS Projects

Research findings on developer and user activities in F/OSS projects mailing lists point out that a small number of individuals are responsible for most of the work. This inequality in contribution raises serious concerns that concentration of development effort on a few will limit knowledge sharing in F/OSS projects [5]. The paradox is that this kind of inequality was observed in many successful projects (eg. Apache, KDE, Friefox).

- **Are Developer and User mailing lists participants doing more posting than replying to questions posited to their lists?** In the Developer list participants contributed more replies (mean=34.52) than posts (7.95). The reverse was the case in the User list. Participants posted more than they replied to questions asked in the list. One explanation for this could be that postings in the developer mailing list may contain sufficient information to allow other participants to analyze the request and are given the highest priority since the

reporter is likely to be a member of the development community. Such posts receive the attention of almost all participants.

- **Is there a trend in the way individuals post and/or reply to questions in Developer and User lists?** About 32.8% (1224) of the participants in the Developer list posted one email message and 19.3% (721) contributed one reply. For the User list 16.1% (963) of the participants posted one email message and 22.9% (1369) contributed one reply. The maximum email messages posted (*nposts*) by one individual was 523 in the Developer list and 4106 in the User list. For the replies, the maximum values were 1517 in the Developer list and 4168 in the User list. In none of the lists did the individual who posted the most emails also made the most replies. For example, in the Developer list, the participant who posted the most emails (523) contributed 574 replies. While the participant with most replies (1517) contributed 79 posts.
- **How are the posts and replies of Developer and User mailing lists participants correlated?** In both lists we found that posting and replying activities of the lists participants are correlated. When posting increases, replying likewise increases. However, the correlation was stronger for the User list (ρ=0.699). This means that, for the User list more questions generate more answers.

6 Summary and Conclusion

A reflection on the empirical investigation revealed inequality in posting and replying activities in the lists studied. Investigating who was asking versus who was replying, we found out that, in the Developer list, the participant who posted the most emails (523) contributed 574 replies. While the participant with most replies (1517) contributed 79 posts. A similar trend was observed in Apache, Mozilla, KDE. What is different in this case is that we looked at knowledge sharing from the perspectives of both developers and users. Yet still, few individuals are involved in the knowledge sharing process. However, while this may be true for one kind of repository (mailing lists), the situation might be different in another kind, bug trackers or version control systems, for instance. An interesting research direction may be to simultaneously study developer and user lists in big and small, successful and unsuccessful projects to see if the pattern of knowledge sharing reported can be generalized. In addition the role of knowledge sharing for open source project success or failure should be better understood and assessed.

References

1. S. K. Sowe, I. Stamelos, I. Samoladas (Eds.) (2007). *Emerging Free and Open Source Software Practices*. Idea Global, May 2007, pp: vi-vii, 98-119.
2. Karl Fogel. (2005). *Producing Open Source Software: How to Run a Successful Free Software Project*, Creative Commons.

3. Crowston, K., Hala, A., and Howison, J. (2003). Defining Open Source Software Project Success. *Proc. of International Conference on Information Systems*, ICIS.
4. Nonaka, I. & Takeuchi, H. (1995). *The Knowledge Creating Company*. Oxford University Press.
5. Kuk, G. (2006). Strategic Interaction and Knowledge Sharing in the KDE Developer Mailing List, *MANAGEMENT SCIENCE*, Vol. 52, 1031-1042.
6. Sulayman K. Sowe, I. Stamelos, L. (2008). Angelis. Understanding Knowledge Sharing Activities in Free/Open Source Software Projects: An Empirical Study, *Journal of Systems and Software*, Vol. 81(4), 431-446.
7. Sheskin, D. J. (2000). *Handbook of parametric and nonparametric statistical procedures*, Chapman & Hall, 491-508.
8. S. K. Sowe. An Empirical Study of Knowledge Sharing in Free and Open Source Software Projects. *PhD thesis*, Department of Informatics, Aristotle University, Greece, 2007.
9. A. Aurum, R. Jeffery, C. Wohlin, and M. Handzic, editors. *Managing Software Engineering Knowledge*. Springer, 2003.
10. S. E. Raymond. The Cathedral and the Bazaar. Musings on Linux and Open Source by an Accidental Revolutionary. OReilly, Sebastopol.