Annals of Knowledge Sharing in Distributed Software Development Environments: Experience from Open Source Software Projects

Sulayman K. Sowe, Rishab Ghosh, and Luc Soete

Collaborative Creativity Group, UNU-MERIT. Keizer Karelplein, 19, 6211 TC Maastricht, The Netherlands
sowe@merit.unu.edu, rishab@dxm.org, luc.soete@algec.unimaas.nl

Abstract: Empirical research aimed at understanding how individuals interact and share their knowledge in a distributed software development environment has traditionally been very difficult because the source of knowledge, the code, has been a guarded secret and software developers and users were seldom in contact, thus making it difficult to study, *in situ*, all the individuals involved in the software development process. Free/Open Source Software (F/OSS) projects presents renewed opportunities, as well as challenges, in understanding collaboration and knowledge sharing amongst software developers and users in distributed software development environments. In this paper, we discuss how developers and users share their knowledge and collaborate in the software development process. Knowledge sharing metrics, software repositories, and suitable methodologies for studying knowledge sharing are presented. The paper aims to stimulate discussion, present our current understanding, and empirical research opportunities and challenges in knowledge sharing in distributed collective practices - F/OSS projects.

1 Introduction

Free and Open Source Software (henceforth F/OSS) has fundamentally changed the way we initiate software projects, develop, distribute, and maintain software. For the first time, the Bazaar model (Re99) provides software engineers an alternative to the Cathedral model or traditional way of developing closed source software. According to the Cathedral model, software development takes place in a centralized way, with well defined roles for each software development phase. In the bazaar model, roles are not clearly defined and usually software users are treated as co-developers. In many projects, developers are also the users of the software they develop. This has a number of advantages including; (i) blurring the developer-user communication gap and facilitating collaboration between all individuals involved in software development, use, and support; (ii) making it easier to study the knowledge sharing activities since project participants are cohabitants of the same project or technology space.

However, we posit that every successful Bazaar styled F/OSS project has some elements of Cathedral style development in it. Some of the differences between the two models are in the areas of user involvement, openness and sharing of software source code, high degree of collaboration between software developers and potential users, informality in the communities involved in various projects, etc. The vibrant F/OSS development process only succeeds because many people volunteer their time and effort to share their knowledge and develop the software. F/OSS projects represent "Distributed Collective Practices" (Tw06) where users are not isolated from direct involvement in the development process as it unfolds. Every member has access to another member and the project's resources; increasing the chance for participants to easily interact and share their knowledge. The success of any project highly depends on the quality of interaction between all participating agents, because this kind of ''*talking to each other*'' is vital for the cooperation and resolution of any conflicts which may arise and the establishment of trust among software development teams (SSA07; ES03).

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. The interaction between complex software infrastructures and the socio-technical aspects of the software can now be studied without organizational barriers. However, our understanding of knowledge sharing in F/OSS projects is challenged by the fact that; (i) coordination and collaboration tools for knowledge sharing can present technological barriers for some users, (ii) individuals or even infrastructures are often located at large distances from each other, which makes exchanging knowledge face-to-face hard, and (iii) the ecology of project's communities is such that people with varying needs, knowledge, and skills participate in the software development process.

Notwithstanding these challenges, empirical studies addressing collaboration and knowledge sharing in software development teams are made easy by the availability of data in software repositories. Researchers can now understand *who* is involved, who is talking to *whom*, *what* is talked about, and the nature of collaboration in the software development process. Our presentation in this paper leverages this opportunity and makes uses of mailing lists data to discuss how software development teams collaborate and share their knowledge in F/OSS projects. Our aim is to provide a platform to discuss and share current understandings and challenges facing empirical research aimed at understanding knowledge sharing, in terms of email exchanges, in open source projects.

The rest of the paper is structured as follows. Section 2 presents background and related work. Section 3 is an empirical investigation which discusses various data sources used by researchers to study collaboration and knowledge sharing in F/OSS projects. Section 4 presents two already tested research methodologies and use live data to show metrics that can be used to study knowledge sharing in F/OSS projects. Research findings, implications, and challenges associated with this kind of empirical study and possible avenues for future research are discussed in Section 5. We draw our conclusion in Section 6.

2 Background and Related Work

The importance of knowledge sharing is not unique to F/OSS projects alone. The knowledge of a software development team in any project is of great value because, according to Steffen and Jorg (SJ08), "it allows [us] to reconstruct at a later time why ideas were discarded, why incompatibilities existed, and how a problem was finally solved". Empirical software engineering is facing the problem that important information held by members of a project can still be tacit and hence difficult to convey to other team members. The problem is how to transform and leverage the tacit knowledge of community members into explicit usable knowledge. Knowledge management (KM) experts often consider how well organizational structures and processes promote collaboration and knowledge sharing (DP00; NT95). The goal of knowledge management should be to lower barriers imposed by organizational and community structures, and geographical boundaries so that project members can generate, archive and share their knowledge. The barriers can be anything from groupware, roles, and activities we need to develop in order to help individuals learn from F/OSS related activities. Daniela, *et al.* (DTE08) defined groupware as a collection of tools, people and work processes operating in synchronized way such that informal tasks are accomplishment in a more effective, efficient and creative way. Lightweight web-based technologies such as the IBM Jazz platform (Lc04), the Eclipse integrated development environment (KG06), TeamSpace, Collaborative Virtual Workspace (CVW), etc. are some tools supporting geographically disperse teams to collaboratively participate and share their knowledge in distributed software development environments.

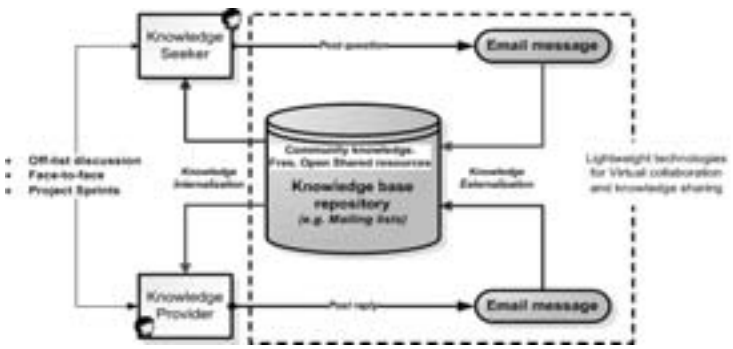

Figure 1. Model for dynamic knowledge sharing.

The F/OSS development process not only highlights a different way of sharing software knowledge but also acquiring knowledge is a complicated process and building on existing knowledge takes time. Software development is a human-based knowledge-intensive activity (Au03; LM03). In addition to sound technological infrastructure (e.g. the Internet) and effective tools (DTE08) to coordinate the collaborative software development process, the success of any project immensely depends on the knowledge and experience brought to it by software developers and users. Collaboration and knowledge sharing process in F/OSS (Figure 1) limits problems associated with knowledge transfer where experience people leave and new developers enter the project.

Figure 1 shows a knowledge sharing model in which knowledge seekers and knowledge providers (SSA06) share their knowledge. According to the model, individuals will exchange ideas and recognize their differences in the perception of a concept. And after multiple iterations of this process, they will come to a mutual agreement. Their collective experiences are archived in a knowledge base repository - a collection of shared and publicly available concepts known as general or public knowledge from which other team members can learn. F/OSS projects are complex environments with many different actors distributed all over the world. There is no indication that all project participants will collaborate and share their knowledge. When members do, a *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;

$$K_{AB}=\{^{1}_{0};\ \textit{where } \boldsymbol{K}_{AB}=1 \textit{ if } A \text{ talks to } B, \textit{ and 0 if otherwise.}$$

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 collaborate with other community members, then he may benefit from their knowledge, from their unique way of doing things and improve his ideas further.

3 Data source for Investigating Knowledge Sharing in Open Source Projects

F/OSS development provides plethora and easily accessible data for research and analysis. Many projects rely on volunteer contributions and extensive peer collaboration through the Internet, using project's mailing lists, *de facto* versioning systems (CVS or SVN), bug tracking systems (BTS) and bug databases (e.g. Bugzilla), Internet Relay Chats (IRC), discussion forums, change logs, etc. These tools not only enable participants to collaborate in the software development process but also act as *software trails* or repositories to store the communication activities of the participants. Researchers can make use of the wealth of information available in these repositories to study collaboration and knowledge sharing in software development teams. For sometime now, these kinds of research has been limited because data was difficult to come by and when available, it is often very small to make valid generalizations. The F/OSS development process partially solves this problem by providing opportunities for researchers to readily obtain large amounts of data from various repositories and be able to combine data from two or more sources in order to carry out cross-fertilization research between projects, communities, and software artefacts. Table 1 shows sources which have provided opportunities to empirically investigate the F/OSS phenomenon on various activity measures including, but not limited to collaboration and knowledge sharing in software development teams. Data sources vary from by-products and enablers of the F/OSS development process (e.g. Bug trackers, Versioning system) to qualitative and quantitative surveys and field reports, to ones resulting from the personal experience and interaction of members of a project in mailing lists, blogs, wikis, etc.

Table 1: Data sources for studying collaboration and knowledge sharing in F/OSS projects.

Data Source	Collaboration and Knowledge Sharing	Researchers
CVS/SVN	Development process, Code evolution, Software quality, developer contribution	(DB05; MFH02; Gd04; BLR04)
Bug Trackers & Change logs	Development process, Code and projects' evolution, Who changed whose code, who commented on what	(GRS04; Ck04; Ca04)
Mailing Lists/ Forums	Knowledge Sharing, Support, help, process, communities, interaction (developer-developer, developer-user, and user-user)	(SSA07; KSL03; LH03)
F/OSS portals	Demographics, landscape, resource usage, Research and infrastructure.	(EST05; RB06; CH03; Xg05)
Surveys, Field studies	Motivation for collaboration & knowledge sharing, developer & Users involvement, economics/concerns of distributed teams	(Rg02; PDS08; HNH03)

These sources provide information about the status of the software and contain 'stamps' indicating who worked on or modified what part of the code, how much lines of code have been added, removed, changed, when changes were done, etc. Tools to support communication and collaboration amongst project participant (e.g. Mailing list, Forums, IRC, etc.) are other sources of data for research (KH05). These sources are important in studying knowledge sharing activities of projects' participants, pattern of communication in projects, formation, structure, composition and evolution dynamics of F/OSS projects and communities. Portals hosting F/OSS projects (e.g. Sourceforge, Savannah, Tigris, etc.) provides data on almost every aspect of the F/OSS development process. In addition, they provide an aggregate of the total number of projects available in a given topic or category. Surveys and field studies are very useful in providing first hand report on what motivate F/OSS participants and users to collaborate and share their knowledge with peers, among other issues.

4 Research Methodologies

Methodologies (Figures 2 and 3) and metrics suitable for investigating knowledge sharing in terms of email exchanges have been proposed (SSA06; Cb06; Kg06). The level of knowledge sharing can be quantified by analysing substantial email exchanges between list participants. This can be achieved by; (i) counting the total number of posts (*nposts* variable) externalized to the list. That is the total number of email messages potential knowledge seekers posted, (ii) counting the total number of replies (*nreplies* variable) made by potential knowledge providers to questions posted to the lists. The value of these two variables provide a simple measure and has been used to investigate knowledge sharing amongst F/OSS projects both in a single and multiple repositories in many projects.

4.1 Studying collaboration and knowledge sharing from a single repository

Research into collaboration and knowledge sharing can be studied from various view points, using as many available data sources as possible. Figure 2 shows how knowledge sharing amongst project's team members may be studied using archived data in mailing lists. The methodology depicts how a researcher can make use of mailing data dumps, with the possibility of other kinds of data from change logs, bug databases, CVS/SVN, etc. Lists archives in the form of *mbox* files can contain data on developers or non-developers. A schema containing the necessary fields can be codified into a script for data mining or directly implemented as database fields in Mysql, for example. The methodology also shows possible data cleaning procedure and tools which can be used to analyse such data.

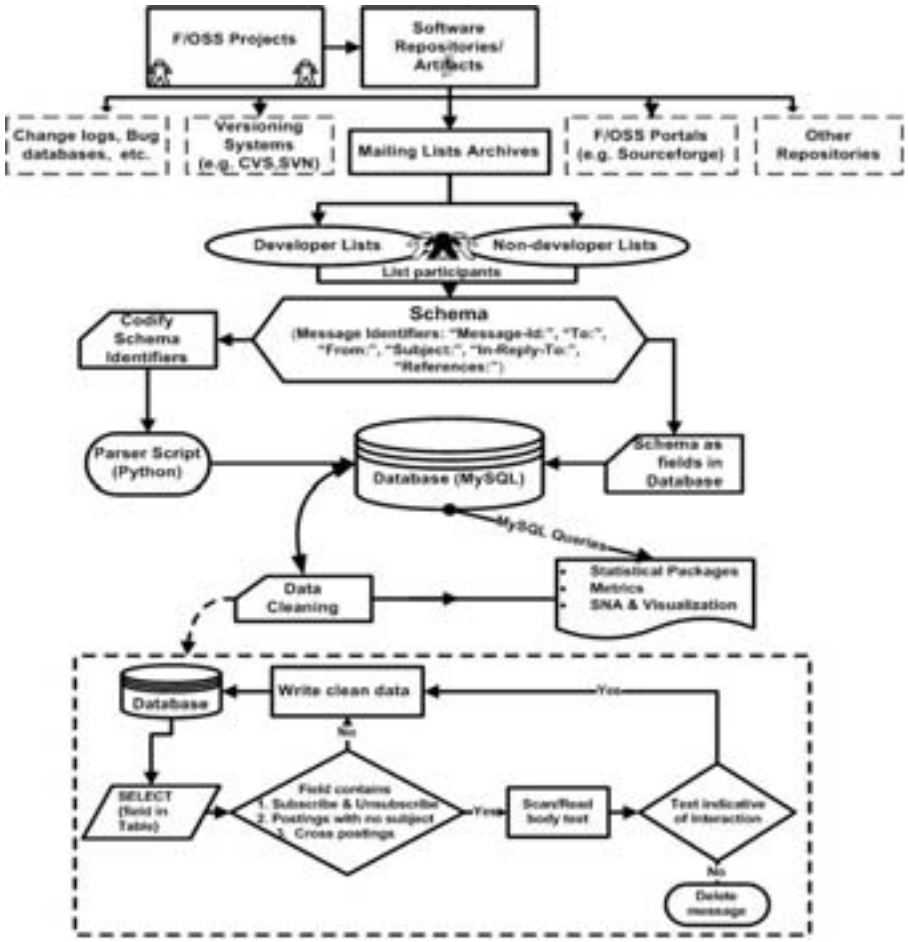

Figure 2. Methodological approach for a single repository (e.g. Mailing lists)

4.2 Studying collaboration and knowledge sharing from multiple repositories

Researchers can also make use of data from more than one repository (Figure 2) to study how distributed software development teams share their knowledge across various repositories; eg. source code/content management systems and mailing lists. The methodology shows the utilization of data from both SVN and mailing lists dumps to study, for example, whether developers are sharing there knowledge more during coding activities in SVN than in their discussion in projects mailing lists [Ss08]. The methodology also shows the possibility to use other data sources such as blogs, wikis, IM, etc. to study collaboration and knowledge sharing in F/OSS projects.

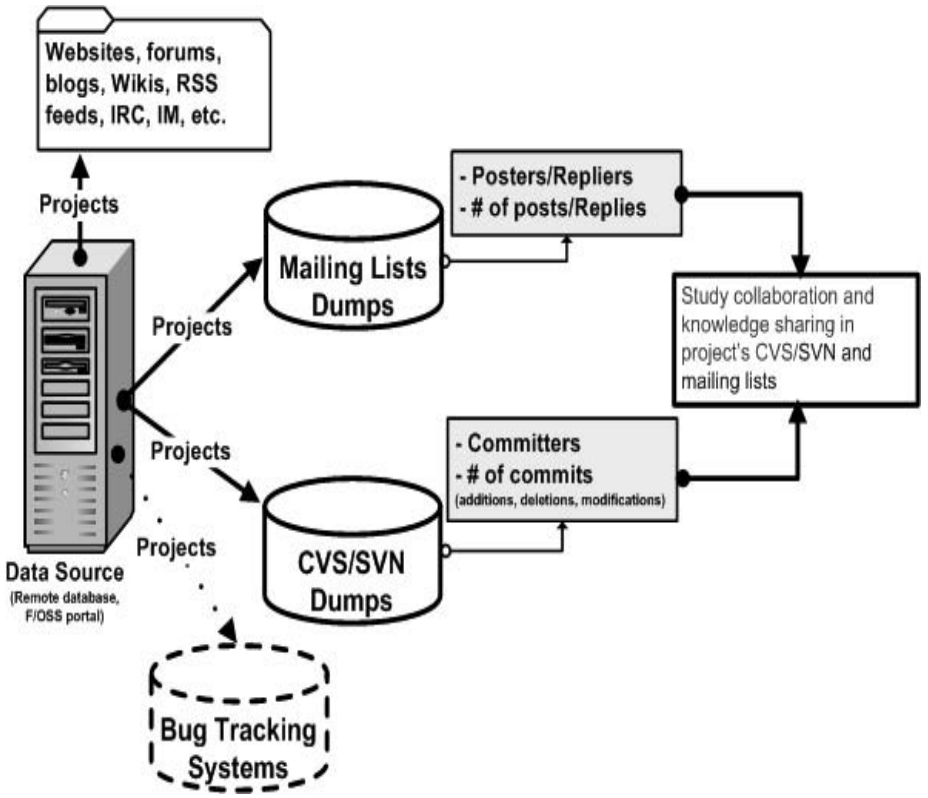

Figure 3: Methodological approach for multiple repositories (e.g. SVN and Mailing lists)

5 Findings, Implications and Challenges

Being it single or multiple repositories, trends are beginning to emerge which shed light on our understanding as to how software development teams collaborate and share their knowledge in a distributed software developments, aka, F/OSS projects.

5.1 The nature of the collaboration and knowledge sharing

The nature of collaboration in F/OSS projects is skewed in all posting and replying activities (figure 4). A small percentage of individuals are responsible for most of the activities. Cliques of developers, usually the core team, intensively collaborate and share their knowledge amongst themselves to muster and drive the project forward. In fact this observation holds for almost all kinds of collaborative activities on the internet, giving rise to the power-law distribution (SSA07, Kg06).

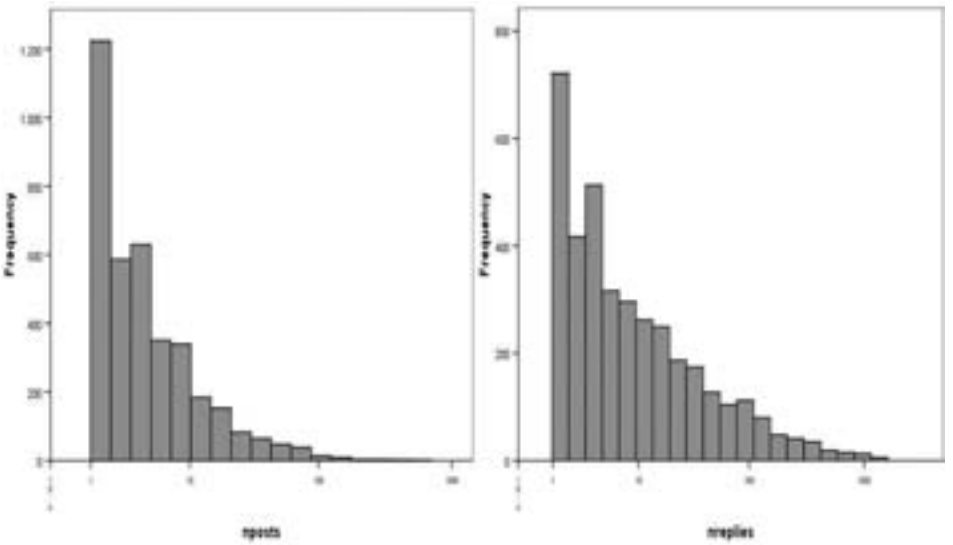

Figure 4: Skewness of posts and replies

A power-law distribution is one in which frequent occurrences of an event (many individuals collaborating) are rarely observed, whereas few incidences of the same event (few individuals collaborating) are very common. Although there is concentration of collaboration and knowledge sharing in the hands of few individuals, in most F/OSS projects anything resulting from the collaborative efforts of few individuals are externalized to the project's forums and lists for the larger community to critique and contribute.

The presence of knowledge brokers (SSA06) have also been identified to play a vital role in diffusing knowledge and expertise across projects and communities. As a measure of knowledge sharing, posting and replying activities in mailing lists are also highly correlated. For example, in the Debian mailing lists, a Spearman's *rho* of 0.608 for the Developer list and 0.699 for the User list was reported (Ss08).

Across multiple repositories, F/OSS developers contribute and collaborate both in their coding activities in SVN and in their developer mailing lists. However, because of the nature of the software development process, developers share their knowledge more in their coding process (Figure 5) and only use mailing lists to communicate their requests, post package information, sort responses and clarification from other users.

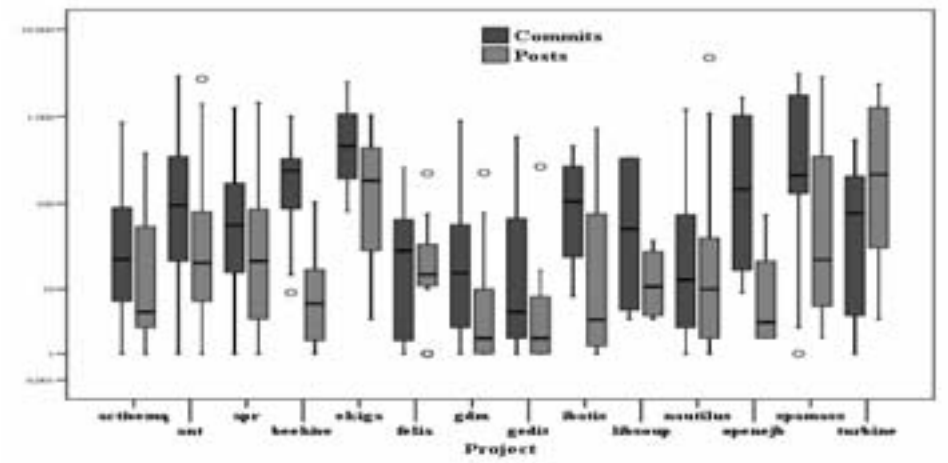

Figure 5. More knowledge sharing in code commits that mailing lists

5.2 Challenges in studying collaboration and knowledge sharing

Software development teams are becoming more and more disperse, crossing geographical boundaries, members coming from different cultures, employing different methodologies and using different tools and technologies at their disposal to accomplish their task. Research aimed at understanding how these teams collaborate and share their knowledge needs to be interdisciplinary; drawing lessons from computer collaborative work, social network analysis, economics, language and anthropology, etc. There is also the need to understand the organizational structures and cultures from which diverse teams may have come from.

Availability of data and metadata in repositories like source control, defect tracking systems, and archived communications helps us understand the nature of collaboration and knowledge sharing in software development team. However, these benefits also come at a cost. We face new and extraordinary research difficulties in dealing with large quantities data which is often unreliable, difficulty to aggregate and filter. Repositories are notorious with noise in their data, largely because of the continuous migration (eg from CVS to SVN or Bugzilla to JIRA) which cause fundamental challenges for researchers. Furthermore, many researchers focus on only on source of data (Ref. Table 1) to study collaboration in software development teams which gives an incomplete picture. A comprehensive understanding of the nature of collaboration and knowledge sharing in software development teams must take into consideration that such teams are not limited to using only one set of tool. Thus, using data from as many repositories as possible to study these teams is much advised.

Current tools allow software development teams to rollback changes, track and record source code and commits history, set continuous integration of builds, see which features are being built, tested, approved, bugsome, etc. However, in order to foster collaboration and knowledge sharing in software development teams, it is vital that these tools, source control and bug trackers in particular, start integrating ontology based *folksonomics,* tagging, semantic web, and social networks technologies in their design architecture. These may enable team members "see" who is who, who can and should do what, and when a certain task should be done and completed.

6 Conclusion

In this paper we have compiled our experience in F/OSS to discuss the dynamics of collaboration and knowledge sharing is distributed software development teams. A knowledge sharing model was introduced to develop understanding of the dynamics of knowledge sharing in F/OSS projects. Our empirical investigation discusses various data sources and metrics used by researchers to study collaboration and knowledge sharing. We presented two research methodologies and used data to discuss the nature of knowledge sharing in F/OSS project.

We have argued that software development teams are diverse and complex. Some of the points we raised in this paper may be very important in helping define the research agenda aimed at comprehensive understanding of collaboration and knowledge in software development teams; (i) diversify and enrich research by studying these teams from many types of repositories, (ii) draw lessons from many fields including economics, sociology, CSWC, etc., (iii) improve design architecture of collaboration and knowledge sharing tools by incorporating semantic web, social software, social networks technologies, and (iv) researchers to start investigating how software development teams collaborate in other mundane tools, which are playing and increasing role in the day-to-day communication between members; IRC, IM, blogs, Wikis.

Acknowledgement

This research is partly funded by the FLOSSMetrics (http://flossmetrics.org/), FP6 Project ID: IST5-033547.

Reference:

[Au03] Aurum, A.; Jeffery, R.; Wohlin, C.; Handzic, M. (Eds.) Managing Software Engineering Knowledge. Springer, 2003.

[BLR04] Jesus. M. Barahona, L. Lopez, and G. Robles. Community structure of modules in the apache project. In Proceedings of the 4th Workshop on Open Source Software Engineering, Edinburgh, Scotland, 2004.

[Ca04] Capiluppi Andrea. Improving comprehension and cooperation through code structure. *e* Proceedings of the 4th Workshop on Open Source Software Engineering, Int. Conf. Software Engineering, May, 2004.

[Cb06] Christian Bird; Alex Gourley; Prem Devanbu; Michael Gertz; Anand Swaminathan. Mining email social networks. In MSR '06: Proceedings of the 2006 international workshop on Mining software repositories, pp: 137–143, 2006.

[Ck04] Chen, K., Schach, S. R., Yu, L., Offutt, J., and Heller, G. Z. 2004. Open-Source Change Logs. Empirical Software Engineering, 9(3), pages 197-210, 2004.

[CH03] Crowston, K., H. A. & Howison, J. Defining Open Source Software Project Success. Proc. of International Conference on Information Systems, ICIS 2003, 2003.

[ES03] Elliott, M. and Scacchi, W. Free Software Development: Cooperation and Conflict in A Virtual Organizational Culture. In S. Koch (ed.), Free/Open Source Software Development, IDEA Press, Chapter 7, Pages 152-172, 2003.

[LH03] Lakhani, K. & Hippel, E. How open source software works: "free" user-to-user assistance. Research Policy, Vol. 32, pages 923-943, 2003.

[Lc04] Li-Te Cheng; Cleidson R.B. de Souza; Susanne Hupfer; John Patterson; Steven Ross. Building Collaboration into IDEs. Queue, 1(9), pp: 40–50, 2004.

[LM03] G. F. Lanzara and M. Morner. The knowledge ecology of open-source software projects. In European Group of Organizational Studies (EGOS Colloquium), Copenhagen., 2003.

[DP00] Davenport, T.; Prusak, L. Working Knowledge. How Organizations Manage What they Know, Havard Business School, Boston, 2000

[DTE08] Daniela de Freitas Guilhermino Trindade;Tania Fatima Calvi Tait1;Elisa Hatsue Moriya Huzita1. A tool for supporting the communication in distributed software development environment. Journal of Computer Science & Technology, 8 (2), pp: 118-124, 2008.

[DB05] Dinh-Trong, T.; Bieman, J. M. The FreeBSD Project: A Replication Case Study of Open Source Development. IEEE Transactions on Software Engineering, Vol. 31, pages 481-494, 2005.

[Gd04] German, D. M. Using software trails to reconstruct the evolution of software. Journal of software maintenance and evolution. Research and Practice, Vol.16(6), pages 367-384, 2004.

[GRS04] Gasser, L.; Ripoche, G.; Sandusky, B. Research Infrastructure for Empirical Science of F/OSS Proceedings of ICSE Workshop on Mining Software Repositories, 2004.

[Gg02] Rishab A. Ghosh; Bernhard Krieger; Ruediger Glott; Gregorio Robles. Free/libre and open source software: Survey and study. Technical report, International Institute of Infonomics, University of Maastricht, The Netherlands, June 2002.

[KG06] Kersten, M; Gail, M. Using Task Context to Improve Programmer Productivity. SIGSOFT, pages 1–11, 2006.

[HNH03] Hertel, G.; Niedner, S. & Herrmann, S. Motivation of Software Developers in Open Source Projects: An Internet-based Survey of Contributors to the Linux Kernel. 2003**.**

[Khh05] Koch S. Hahsler M. Discussion of a large-scale open source data collection methodology. In Proceedings of the 38th Hawaii International Conference on System Sciences (IEEE, HICSS '05-Track 7), Jan 03-06, Big Island, Hawaii, page 197b., 2005.

[Kg06] Kuk, G. Strategic Interaction and Knowledge Sharing in the KDE Developer Mailing List. Management Science*,* Vol. 52, pages 1031-1042, 2006.

[KSL03] Krogh, G.; Spaeth, S. Lakhani, K. Community, joining, and specialisation in open source software innovation: a case study. Research Policy*, 32*, pages 1217-1241, 2003

[MFH02] Mockus, A.; Fielding, R.; Herbsleb, J. Two case studies of open source software development: Apache and Mozilla. Transactions on Software Engineering and Methodology, Vol. 11(3), pages 1-38, 2002.

[SJ08] Steffen Lohmann and Jorg Niesenhaus. Towards Continuous Integration of Knowledge

Management into Game Development. Proceedings of I-KNOW '08 and I-MEDIA '08 Graz, Austria, September 3-5, 2008.

[NT95] Nonaka, I.; Takeuchi, G. The Knowledge Creating Company. Oxford University Press., 1995.

[PDS08] Panjer, L. D., Damian, D., and Storey, M. 2008. Cooperation and coordination concerns in a distributed software development project. In Proceedings of the 2008 international Workshop on Cooperative and Human Aspects of Software Engineering, Leipzig, Germany, May 13 - 13, 2008.

[Re99] Raymond, E. S. The cathedral and the bazaar. Fristmonday, 3(3), 1999.

[RB06] Robles, G. and Barahona, J. Geographical Location of Developers at SourceForge *MSR 2006, Shanghai, China,* 2006.

[SSA06] Sowe, S. K.; Stamelos, I.; Angelis, L. Identifying knowledge brokers that yield software engineering knowledge in oss projects. Information and Software Technology, Vol. 48, pages 1025–1033, 2006.

[SSA07] Sowe, S. K.; Stamelos, I.; Angelis, L. Understanding knowledge sharing activities in free/open source software projects: An empirical study. Journal of Systems and Software, Vol. 81(3), pp: 431-446, 2008.

[Sow08] Sulayman K. Sowe, Ioannis Samoladas, Ioannis Stamelos, Lefteris Angelis. Are FLOSS developers committing to CVS/SVN as much as they are talking in mailing lists? Challenges for Integrating data from Multiple Repositories. 3rd International Workshop on Public Data about Software Development (WoPDaSD). September 7th - 10th 2008, Milan, Italy.

[EST05] Elliott S.; Singer J.; Timothy, L. Studying software engineers: Data collection techniques for software field studies. Empirical Software Engineering, Vol. 10, pages 311–342, 2005.

[Tw06] Turner, William; Bowker, Geoffrey; Gasser, Les; Zacklad, Manuel. Information infrastructures for distributed collective practices. Computer Supported Cooperative Work, Vol.15(2-3), pages 93–110, 2006.

[Xg05] Xu, J.; Gao, Y.; Christley, S. & Madey, S. A topological Analysis of the Open Source Software Development Community. *I*EEE Proceedings of the 38th Hawaii International Conference on System Sciences, (IEEE, HICSS '05-Track 7), Jan 03-06, Big Island,Hawaii., 2005, 198a.