Available online at www.sciencedirect.com
ScienceDirect
The Journal of Systems and Software xxx (2007) xxx–xxx
The Journal of Systems and Software
www.elsevier.com/locate/jss

Understanding knowledge sharing activities in free/open source software projects: An empirical study

Sulayman K. Sowe *, Ioannis Stamelos, Lefteris Angelis

Aristotle University, Department of Informatics, 54124 Thessaloniki, Greece

Received 15 August 2006; received in revised form 15 March 2007; accepted 25 March 2007

Abstract

Free/Open Source Software (F/OSS) projects are people-oriented and knowledge intensive software development environments. Many researchers focused on mailing lists to study coding activities of software developers. How expert software developers interact with each other and with non-developers in the use of community products have received little attention. This paper discusses the altruistic sharing of knowledge between knowledge providers and knowledge seekers in the Developer and User mailing lists of the Debian project. We analyze the posting and replying activities of the participants by counting the number of email messages they posted to the lists and the number of replies they made to questions others posted. We found out that participants interact and share their knowledge a lot, their positing activity is fairly highly correlated with their replying activity, the characteristics of posting and replying activities are different for different kinds of lists, and the knowledge sharing activity of self-organizing Free/Open Source communities could best be explained in terms of what we called "Fractal Cubic Distribution" rather than the power-law distribution mostly reported in the literature. The paper also proposes what could be researched in knowledge sharing activities in F/OSS projects mailing list and for what purpose. The research findings add to our understanding of knowledge sharing activities in F/OSS projects.

Keywords: Open source software projects; Mailing lists; Knowledge seekers; Knowledge providers; Knowledge sharing; Open source communities; Power-law distribution; Self-organizing communities

1. Introduction

The user of Free and Open Source Software (F/OSS), having access to the source code, is free to study what the program does, modify it to suit his/her needs, distribute copies to other people and publish improved versions so that the whole F/OSS community can benefit. The licenses agreement (e.g. the General Public License or GPL) under which the source code is distributed defines exactly the rights the user has over the product. In the literature, many terms are in use to describe the F/OSS phenomenon. Notably, Free Software (FS), a term used by Free Software Foundation (FSF) and Open Source Software (OSS) used by the Open Source Initiative (OSI). In addition, Free Open Source Software (FOSS), Free/Libre/Open Source Software (FLOSS), and Libre Software (LS) are terms frequently used by researchers. Without going further into the terminology or ideological differences, in this paper we use F/OSS, like in Koch (2004), to acknowledge the inspiring work done by both the FSF and OSI.

The F/OSS development process exemplifies a viable software development approach. It is also a model for the creation of self-learning (Sowe et al., 2004; von Krogh et al., 2005) and self-organizing communities (Valverde et al., 2006; Sowe et al., 2005) in which geographically distributed individuals contribute to build a particular application by means of the *Bazaar model* (Raymond, 1999). The model has produced a number of successful applications in the area of operating systems (Linux), emailing and web services (Gmail, Apache), databases (MySQL,

* Corresponding author. Tel.: +30 231091927; fax: +30 2310998419.
E-mail address: sksowe@csd.auth.gr (S.K. Sowe).

0164-1212/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2007.03.086

PostgreSQL), etc. The success of these products has changed the ecology and dynamics of F/OSS communities. Large numbers of technical and non-technical end-users are participating in F/OSS projects (Fitzerald, 2004; Nichols and Twidale, 2003). They get involved in activities that are essential for the F/OSS development process (Fitzerald, 2004), as well as the maintenance and diffusion of the software (Lakhani and Hippel, 2003; Michlmayr, 2004). Project activities may include user support (Fitzerald, 2004), suggesting new features (Krogh et al., 2003), testing and debugging software (Sowe et al., 2006), etc. In order to understand the nature of these activities, many researchers focused on mailing lists in conjunction with source code repositories (German, 2004; Koch and Schneider, 2002; Krogh et al., 2003; Lakhani and Hippel, 2003). These studies provide great insight into the collaborative software development process that characterizes F/OSS projects. However, many aspects of community participation in F/OSS projects is still not fully understood (Healy and Schussman, 2003; Valverde et al., 2006; Michlmayr, 2004; Scacchi, 2006; Schofield and Cooper, 2006), especially when it comes to knowledge sharing activities (Kim, 2003; Lanzara and Morner, 2003; Sowe et al., 2006). Yet, F/OSS is characterized as intensely people-oriented (Timothy et al., 2005) and knowledge intensive software development process (Krogh et al., 2003; Lanzara and Morner, 2003). F/OSS projects are complex cognitive systems (Sowe et al., 2006) and, as Ye and Kishida (2003), Ye et al. (2004) pointed out, the knowledge needed for the software development process is vast and unlikely to be held by one or a small group of software developers. Thus, understanding the nature in which knowledge is generated, archived, and shared by community members is vital if we are to consolidate and increase our understanding of the software development process. Our contribution in this paper is towards understanding knowledge sharing activities in F/OSS projects and the implications such activities may have on the software development process. The lists we selected for our study are the *Developer* and *User* mailing lists of the Debian project. Developer lists are traditionally where software developers discuss core software development activities and have been found to generate little discussion (Healy and Schussman, 2003; Kim, 2003; Krishnamurthy, 2002; Mockus et al., 2002). User lists, on the other hand, are mostly frequented by individuals or project participants who need help on various issues related to F/OSS. By studying the knowledge sharing activities of participants in these two kinds of lists, we hope to have a wider coverage and an in-depth understanding of how F/OSS project participants share their knowledge. First, we give the definition of terms we shall be using in this study.

1.1. Definition of terms

The main focus of our research is knowledge sharing between knowledge providers and knowledge seekers. We define a *knowledge provider* in F/OSS projects as an expert software developer who helps project participants on various issues related to software development and use. Any list participant who seeks assistance on issues related to software development and use (e.g. how to compile code, run an application, configuration details, resolve package dependencies, documentation, etc.) can be described as a *knowledge seeker*. In F/OSS projects, developers are themselves users of the software. Most mailing lists are open to all participants so that users can ask questions, and developers can post patches for others to review. Sometimes a software developer or module maintainer may assume the role of a knowledge seeker by posting to lists, asking questions relating to software configuration, package dependency issues, bugs, etc. At the same time, an ordinary software user may assume the role of a knowledge provider by answering questions others ask in the lists. Roles are not assigned in F/OSS projects, almost every activity is voluntary (Michlmayr, 2004). Depending on one's expertise, anyone can assume any role in the project. Thus, the distinction between a knowledge seeker and a knowledge provider depends only on the unanticipated role of the individual at a particular moment in time.

Our view *of knowledge sharing* in F/OSS projects is in agreement with Zeldin (1999). In this view, sharing knowledge is a synergistic process – "you get more out than you put in." If a mailing list participant shares his ideas or a way of installing or configuring particular software with another person, then just the act of putting his idea into words will help him shape and improve that idea. If he enters into a dialogue with the other mailing list participants, then he may benefit from their knowledge, from their unique way of doing things and improve his ideas further. Each list participant enters into a 'conversation' with other participants in the list. When two or more participants exchange email messages, they are said to share their knowledge. Knowledge sharing in F/OSS projects is all about helping each other and collaboration. The F/OSS development context is a symbiotic cognitive system, where the community learns from its participants, and each individual learns from the community (Sowe et al., 2005). 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 mailing from which subsequent participants can learn.

In the first part of our research we analyze the posting and replying activities of the participants in each list by counting the number of email messages they posted and the number of replies they made to questions others posted. The distributions of posts and replies are then analyzed and compared using various non-parametric measures. Second, considering that the distributions of many F/OSS activities are skewed in nature (Xu et al., 2005; Koch, 2004; Hunt and Johnson, 2002; Wu and Holt, 2006), we test to see if the distribution of the posts and replies in the lists obey the power-law. The rest of the paper is structured as follows. Our theoretical foundation in Section 2 is aimed at understanding knowledge sharing dynamics in F/OSS projects' mailing lists and the metrics for measuring knowledge sharing. In Section 3, we give an overview of relevant

Please cite this article in press as: Sowe, S.K. et al., Understanding knowledge sharing activities in free/open ..., J. Syst. Software (2007), doi:10.1016/j.jss.2007.03.086

prior work and present our research questions. The research methodology, sampling method, data collection and cleaning techniques we used are presented in Section 4. In Section 5, we report on our data and discuss the results. Section 6 summarizes our findings, validity threats, and future research. The conclusion to our research is presented in Section 7.

2. Theoretical foundation

The aim of our theoretical foundation is to understand knowledge sharing dynamics in mailing lists. In most F/OSS projects mailing lists are the preferred means for coordinating software development and support activities. Ongoing interactions between project participants are a means of acquiring valuable software knowledge that is worth archiving. Mailing lists play a vital role in connecting knowledge seekers who are searching for knowledge with knowledge providers who already posses this knowledge. Through mailing lists, software users can ask questions and get answers. Software developers can discuss code development; package maintainers can disseminate product updates, get feedbacks, and discuss bugs and software dependencies. The Knowledge Sharing Model (KSM) in Fig. 1 shows how mailing list participants share their knowledge by exchanging one or more email messages. Their knowledge and concepts are archived into the project's mailing list or Knowledge Base (Sowe et al., 2005). This is a collection of shared and publicly available artifacts known as reusable or public knowledge. Other repositories include Concurrent Versions Systems (CVS), Subversion (SVN), Frequently Asked Questions (FAQs), project web sites, bug databases (Lanzara and Morner, 2003), etc. Knowledge seekers and/or knowledge providers transfer their knowledge, expertise, or know-how to the mailing list by means of a process called *externalization*. The process of acquiring knowledge from the mailing list and filtering of that knowledge to provide greater relevance to the acquirer is called *internalization* (Sowe et al., 2005).

As directions of the arrows in Fig. 1 show, a potential knowledge seeker composes a message by posting or externalizing it to the list (**A**). A knowledge provider consults the list and internalizes that knowledge (**B**). He may reply if the post interests him (**C**). At a later date, the knowledge seeker revisits the post and internalizes the knowledge (**D**). For example, a potential knowledge seeker confronts an unfamiliar concept (or bug) in the use of an application (e.g. OpenOffice) and decides to seek help from the project's mailing list. There are two ways to look at this scenario in the KSM:

(1) *If the concept has been encountered before*, it will be captured and stored in the knowledge base in the form of threaded discussions, which represent software knowledge resulting from interaction between list participants. This knowledge is externalized into the project's mailing list, indexed and archived, for subsequent knowledge seekers and knowledge providers to utilize by internalization. The knowledge seeker then directly consults the list.
   a. Knowledge seeker internalizes knowledge from knowledge base: (**KB**) → (**D**). According to this model, software experts or knowledge providers are also continuously browsing the mailing list to seek knowledge.
   b. Knowledge provider internalizes knowledge from knowledge base: (**KB**) → (**B**).

   The source of knowledge in both cases is the mailing list.

(2) *If the concept has not been encountered*, this means that the knowledge seeker's problem has not been addressed in the mailing list before. He then identifies the appropriate list, posts his question, and exchanges ideas with list participants.

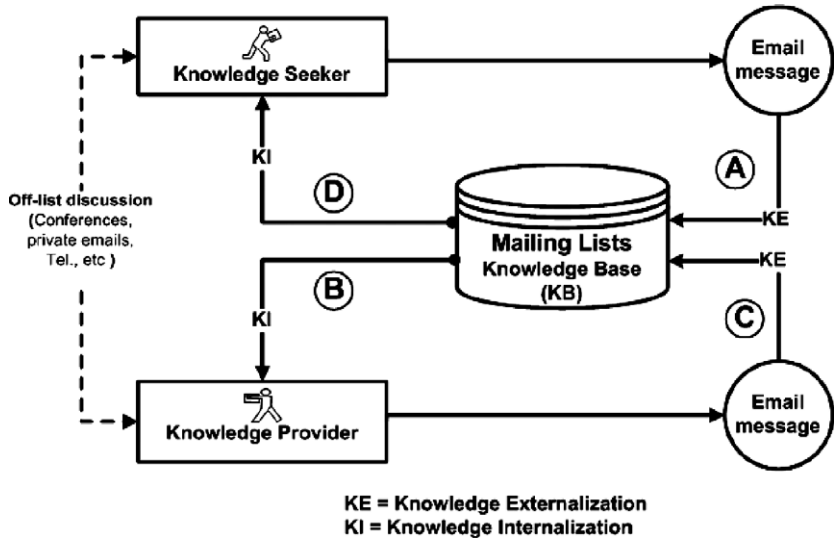

Fig. 1. Knowledge sharing model.

a. Knowledge seeker posts his question to the list: (**A**) → (**KB**).
b. Knowledge provider may browse the list: (**KB**) → (**B**).
c. Knowledge provider decides to reply to the list: (**C**) → (**KB**).
d. Knowledge seeker internalizes knowledge from knowledge base: (**KB**) → (**D**).

Alternatively, the knowledge seeker may know a participant with expertise in the area he is interested in and exchange direct emails with that person (shown in dotted lines), with or without copies being posted to the list.

2.1. Process of measuring knowledge

When talking about knowledge, a difference is made between tacit and explicit knowledge. Tacit knowledge is more difficult to code, articulate, and transfer since it is often deeply rooted in the owner's head. It has been argued that tacit knowledge is subconsciously understood and applied, developed from direct experience and action (Zack, 1998). Explicit knowledge, in contrast, can be expressed or articulated into signs, text, and words. It can be more easily codified and documented (Davenport and Prusak, 2000). As such, explicit knowledge is transmitted and shared in F/OSS via the Internet and through personal communication means (direct exchange of emails between participants). In this context, the F/OSS participant must interact with the entities (e.g. other participants in lists, websites, forums, to-do lists, etc.) in which explicit knowledge is contained to have an understanding of what the project is all about (Sowe et al., 2005). However, it does not follow that another project participant can comprehend and correctly value the knowledge due to differences in programming capabilities or experience in the project.

The process of measuring knowledge is complicated by its intangible nature (Atreyi and Bernard, 2004), especially measuring tacit knowledge. But when tacit knowledge is transformed into explicit knowledge through socialization or interaction (Nonaka and Takeuchi, 1995) and shared by members of an organization or project, an attempt can be made to measure how much knowledge is being shared. Koh and Kim (2004), suggested a way we can quantify the level of knowledge sharing in virtual communities. Similarly, we can provide quantifiable measures of knowledge sharing activities in F/OSS projects' mailing lists by analyzing substantial email exchanges between list participants. We accomplish this in our KSM model by

- 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.
- 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.

These two values will provide a simple measure we can use to quantify knowledge sharing in the lists.

3. Relevant prior work and research questions

Many researchers have investigated what motivates software developers to participate in F/OSS projects in general (Hertel et al., 2003), and their coding activities in particular (German and Mockus, 2003; Koch and Schneider, 2002; Scacchi, 2006; Koch, 2004). But too little is known about how software developers or users share their knowledge in their respective projects mailing lists. Studies utilizing developer mailing list show that only a handful of core and active developers discuss code development (Barahona et al., 2004; Mockus et al., 2002) and evolution (German, 2004). Some of these lists (e.g. in the Apache project) generate little discussion (Mockus et al., 2002). Few studies have investigated how software experts share their knowledge and support non-technical end-users. For example, Lakhani and Hippel (2003) utilized the Apache 'field support system' to study how information [knowledge] seekers post their questions, and potential information [knowledge] providers read and post answers. They found that about 2% of the knowledge providers were responsible for about 50% of the answers to questions posted on the help system and 50% of the questions were provided by 24% of the knowledge providers. The 100 most active information seekers posted an average of 10.43 questions and the 100 most active information providers posted an average of 83.63 answers during the 4-year period of their study. This means that only few individuals are active in providing answers to questions asked in the Apache system. This has implications for the nature of knowledge sharing. For example, what happens when active knowledge providers resign form participating in the help system? Will someone step in to provide answers to questions posited to the list? If some of these knowledge providers are also active package maintainers or core programmers, how do their activities in the lists affect their programming productivity? In another study of the interaction between developers and users in the TouchGraph and SquirrelMail projects' mailing list (Kim, 2003) found strong similarities in the types of interaction that occur on their mailing lists; almost equal number of participants posting and replying to questions in the lists. In both of the latter studies there was no methodology to tell us how the individuals or their contributions were identified. For example, Kim (2003) counted the posts individuals contributed to the forums of the two projects studied. But it was not clear whether the posts originated from the poster (with no 'Re:' in the subject header) or whether the posts were replies (with 'Re:' in the subject) to other posts made earlier to the list. Krogh et al. (2003) investigated how individuals join the Freenet project's mailing list using 'joining scripts' and how newcomers interact with professional software developers. On the developer list they found that 36.7% of new participants

Please cite this article in press as: Sowe, S.K. et al., Understanding knowledge sharing activities in free/open ..., J. Syst. Software (2007), doi:10.1016/j.jss.2007.03.086

who would like to join as coders did not get any response. This implies that, in some projects, a reasonable number of knowledge seekers will not have answers to their questions. While the study is important in highlighting the joining of individual newcomers or developers, the knowledge sharing aspect of the overall Freenet community was not addressed. The social network visualization technique employed in Sowe et al. (2006) identified knowledge brokers in three Debian mailing lists but did not provide quantifiable measures of the postings and replies of the lists participants. Knowledge brokers, according to Sowe et al. (2006), serve an important role in mailing lists as community facilitators, 'moving' from one list to another helping answer questions knowledge seekers posted.

3.1. Research questions

The aforementioned studies show that our understanding of knowledge sharing activities in F/OSS projects is by no means complete. In some projects few individuals contribute most of the posts (Lakhani and Hippel, 2003), while in some projects equal numbers of mailing list participants are involved in posting questions and replying to questions asked in the lists (Kim, 2003). Further still, in some projects many knowledge seekers will not have their questions answered (Krogh et al., 2003). What is more, there is no methodology to identify who is doing the posting and who is doing the replying in mailing lists. Our goal in this paper is to explore these issues and provide answers to some questions which will help us understand knowledge sharing activities of F/OSS projects' mailing lists.

Q1. Are Developer and User mailing lists participants doing more posting than replying to questions posted to their lists?
Q2. Is there a trend in the way individuals post and/or reply to questions in Developer and User lists?
Q3. How are the posts and replies of Developer and User mailing lists participants correlated?

To answer these questions we shall analyze the posting and replying activities of 3735 participants in the Developer list and 5970 participants in the User list. In our analysis we used various statistical measures to help us understand knowledge sharing activities in the Debian project. In our introduction, we discussed F/OSS communities as self-learning and self-organizing. Each mailing list can be viewed as 'sub-community' of the overall project's community. In sharing their knowledge, mailing lists participants form a self-organizing community – a loose association of people who have a common interest in helping each other, developing, testing, improving, and using the software. These activities culminate in a number of posts and replies. We are also interested in answering the question.

Q4. How can the knowledge sharing activities of Developer and User mailing lists participants be explained in terms of the self-organizing structure of F/OSS communities?

For this question we used power-law scaling to find out whether there exists linear correlation between a participant's rank and his contribution to knowledge sharing in the list.

4. Methodology and data

Compared to traditional research practices under proprietary software, F/OSS development provides researchers with an unprecedented abundance of easily accessible data for research and analysis. A huge amount of data is available to study community participation in F/OSS projects (Barahona et al., 2004; German and Mockus, 2003; Ghosh, 2004; Hahsler and Koch, 2005; Koch and Schneider, 2002) and developers and users involvement in projects mailing list (Krogh et al., 2003; Lakhani and Hippel, 2003; Sowe et al., 2006). Our data collection, extraction, and cleaning methodology are shown in Fig. 2.

4.1. Sampling and data collection

Our study utilized data from the Debian project lists archives. Debian was selected because the project provides opportunities for researchers to observe F/OSS community participation (Barahona et al., 2004; Michlmayr, 2004; Sowe et al., 2006). The Debian project hosts over 100 lists on all aspects related to the project. From the Debian lists archives (Debian Mailing Lists) we selected two high volume mailing lists. The following lists are analyzed in our study:

- *Debian-user*. This list is specifically dedicated to help and discussion among users of Debian who speak English.
- *Debian-devel*. This list is specifically dedicated to discussion about technical development topics.

Our data collection period for both lists was from January 2000 to December 2005. We obtained archived mbox files of the two lists. Each file is a single text file containing one month of archived email messages. Every email message has a unique message-id, together with other identification fields defined by the Internet Message Format (RFC) 2822 (Internet Message Format, 2001).

4.2. Data extraction

The coding schema shown in Table 1 was developed to extract the message identifiers and map them as fields in a database:

A Python script implementing the schema was used to extract data from the mbox files. For a given input list, the script traversed each mbox to extract a record for each *msg_id* (primary key). The output for each run was parsed into a MySQL database containing two tables, one for

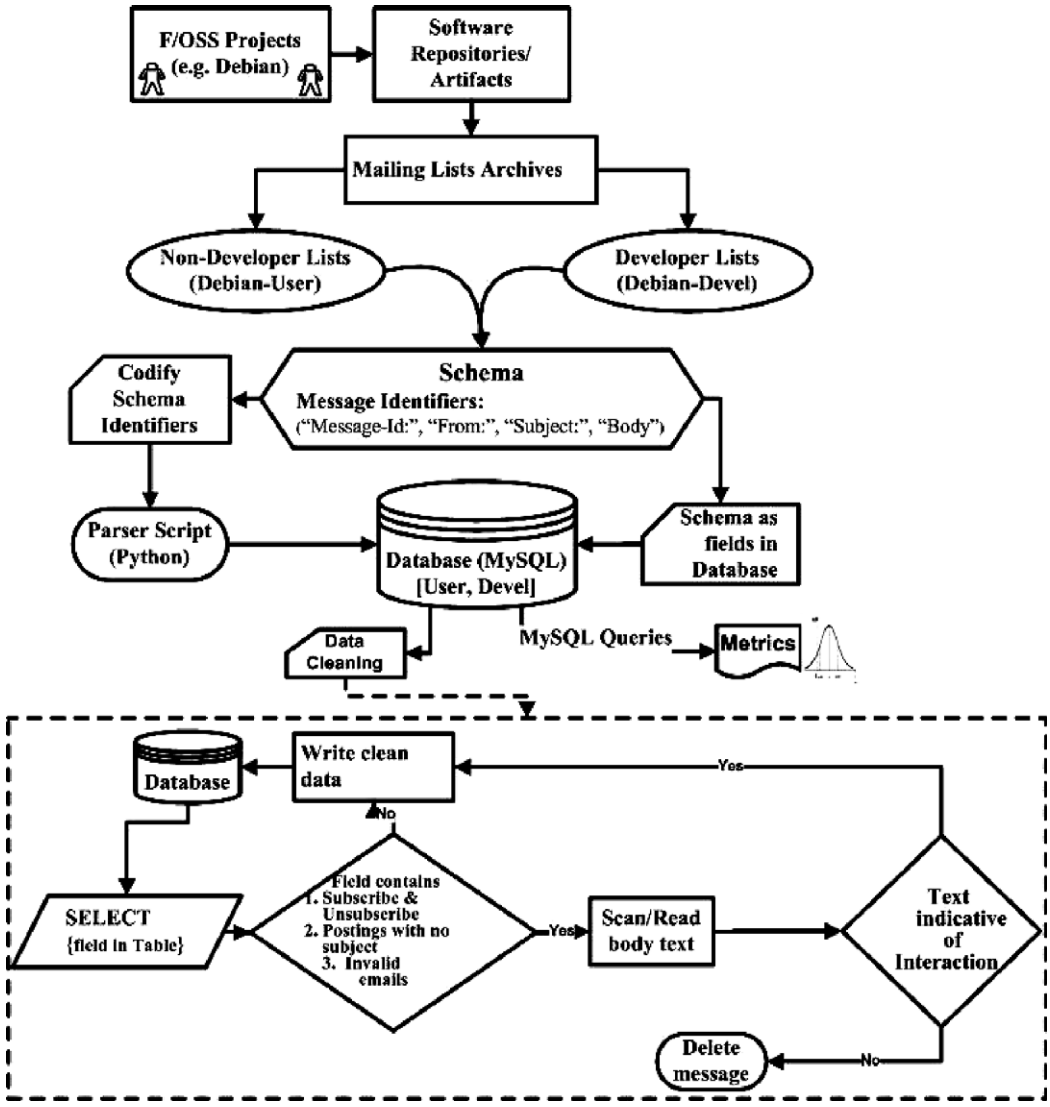

Fig. 2. Methodological outline. Modified from Sowe et al. (2006, p. 1027).

Table 1
Data extraction schema

mbox (RFC) format	MySQL database field	Interpretation
Message-ID:	*msg_id*	Uniquely identifies a message
From:	*sender*	Origin or poster of a message
Subject:	*subject*	Subject header of a message
Plain text	*msg_body*	Message content

each list. SQL queries were used to extract the information necessary for data analysis.

4.3. Data cleaning

It is becoming increasingly evident that collecting and analyzing F/OSS data has become a problem of abundance and reliability in terms of storage, sharing, aggregation, and cleaning (Conklin, 2006; Sowe et al., 2007). In order to improve the quality of our data and account for participants' interaction, certain messages had to be removed. The data cleaning process involved removing messages in the following categories:

Subscribe and unsubscribe messages: The two tables were queried for email messages with 'subscribe' and 'unsubscribe' in the 'Subject' field. Postings in this category were inspected and removed.
Postings with no subject: We needed the 'subject' field to identify whether a particular message is an initiated post or a reply. We queried the tables for 'empty' subject headers and carried out further cleaning to ensure that the same sender did not repost the same message with a subject. The outputs were inspected and messages in this category were removed.
Invalid email addresses: We found some Email addresses with invalid characters. For example, emails of the

type "?\''{A}??'@msj.debian.org"' or "..@mx", were removed.

4.4. Identification of list participants

From the data two types of individuals could be identified; posters and repliers.

Posters: A poster is a participant who initiates or posts at least one email message to a list. The initiated post has no "Re:" in the subject header. The "sender" field in the database identifies a poster and contains two elements – the poster's name (first and last name) and his email address. The name was used to track whether the same person used different email addresses. The tracking was implemented by writing the contents of the "sender" field into another table containing two fields (name|email). The table relates the names with their corresponding email addresses. In this way, we could identify emails belonging to the same poster. Where the same poster used different emails, we counted him as one.

Repliers: A participant is identified as a replier when he/she sends a message which has "Re:" in the "Subject" field. While other message identifiers such as "In-Reply-To:" and "References:" could be used to identify replies and/or repliers Gloor et al. (2003), a study by Sowe et al. (2006) established that messages with "Re:" in the "Subject" field were the best means to identify repliers.

5. Results and discussions

The data we extracted from each of the lists has three items worth considering; the email or identity of the individual, total emails he posted, and total replies he made. We removed all individuals who only posted emails and made no replies and vise versa; i.e., 0 in the number of posts or replies. This was necessary in order to account for knowledge sharing between the participants. Every knowledge seeker who posted an email message also posted a reply as a knowledge provider. Thus, in our data each participant made at least one post and one reply. The status of our data before and after cleaning is shown in Table 2. From the table it can be seen that much effort is required in cleaning the User list's data. This may be due to the fact that it's a high volume list and open to everyone, novice and experts alike.

5.1. Posting and replying activities of mailing lists participants

For the posting and replying activities of lists participants we used the two variables (*nposts* and *nreplies*) to denote the number of emails messages posted and the number of replies made by an individual. From the data we collected over a 5-year period, the 3735 participants in the Developer list posted 29,685 email messages, which generated 128,933 replies. The 5970 participants in the User list posted 193,276 email messages, which generated 765,380 replies. Table 3 shows the descriptive statistics of the data for the two lists. Comparatively, the mean (posts/person) value and the median of *nposts* are smaller for the Developer list. The mean and median for *nreplies* are larger for the Developer list. Both variables have largest maximum value in the User and also largest standard deviation, skewness and kurtosis. Furthermore, while the central tendency (mean, median) of *nposts* is larger in the User list, the central tendency of *nreplies* is larger in the Developer list.

The histograms in Figs. 3 and 4 (x-axis in logarithmic scale), and the box-plots in Fig. 5 (y-axis in logarithmic scale) shows how the posts and replies are distributed in each list.

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.

Furthermore, we applied non-parametric tests to show the difference between the distributions of the posts and replies. The Mann–Whitney and the Kolmogorov–Smirnov tests we used gave $p = 0.000$ (<0.001) for both *nposts* and *nreplies* between the two lists, showing statistically significant difference in the posting and replying activities. This

Table 2
Nature of data before and after cleaning

Status of data	Developer list			User list		
	Individuals (Posters/repliers)	Posts *nposts*	Replies *nreplies*	Individuals (Posters/Repliers)	Posts *nposts*	Replies *nreplies*
Before cleaning	9869	29,721	129,836	11,109	220,187	876,214
After cleaning	3735	29,685	128,933	5,970	193,276	765,380
Discrepancies	6134	36	903	5139	26,911	110,834

Please cite this article in press as: Sowe, S.K. et al., Understanding knowledge sharing activities in free/open ..., J. Syst. Software (2007), doi:10.1016/j.jss.2007.03.086

Table 3
Descriptive statistics of the data for the two lists

		List	
		Developer	User
nposts	Mean	7.95	32.37
	Median	3.00	7.00
	Std. Dev.	21.302	121.753
	Minimum	1	1
	Maximum	523	4106
	Skewness	11.411	15.972
	Kurtosis	199.980	385.529
nreplies	Mean	34.52	27.70
	Median	6.00	5.00
	Std. Dev.	105.567	122.040
	Minimum	1	1
	Maximum	1517	4168
	Skewness	7.061	17.281
	Kurtosis	63.256	434.102

can better be seen in the box-plots in Fig. 6. We divided all the data in three categories according to the *nposts*

a. *nposts* = 1–10 (a small number of posts),
b. *nposts* = 11–100 (medium number of posts), and
c. *nposts* > 100 (a large number of posts).

When we superimposed the posts and replies of the two lists, the difference between the posting and replying activities of the participants in the two lists was clearly visible, as shown in Fig. 7. The linear plot on the left-hand side of Fig. 7 shows that most participants in the User list contributed small posts and small replies. Few of the counts of posts and replies were above the 200 mark. For the developer list, participants' activities were characterized by a small number of posts and a large number of replies. The plot on the right of Fig. 7 is in logarithmic scale.

5.2. Relationship between participants' posting and replying activities

Correlations and regression analysis were used to study the relationship between the posts and replies in the lists. We report on three types of correlations. The Pearson values give us an idea of the linear correlation between the *nposts* and *nreplies* variables. Indeed, the Kendall's *tau_b* and Spearman's *rho* are better in our case, since they work with ranks and not original values. Using the original values of our data, Pearson's *r* for the Developer and User lists were 0.518 ($p = 0.000$) and 0.926 ($p = 0.000$), respectively. The other two correlations, computed by rank, for

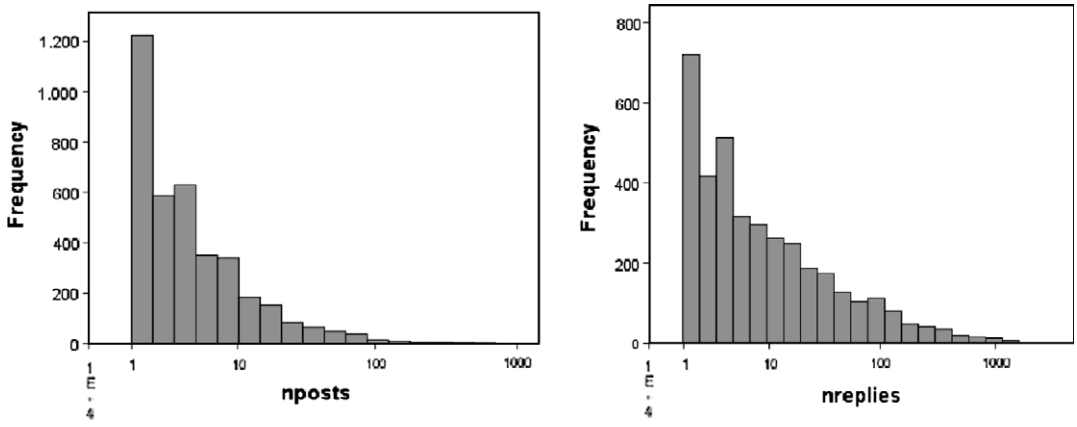

Fig. 3. Frequency distributions of posts and replies in the Developer list.

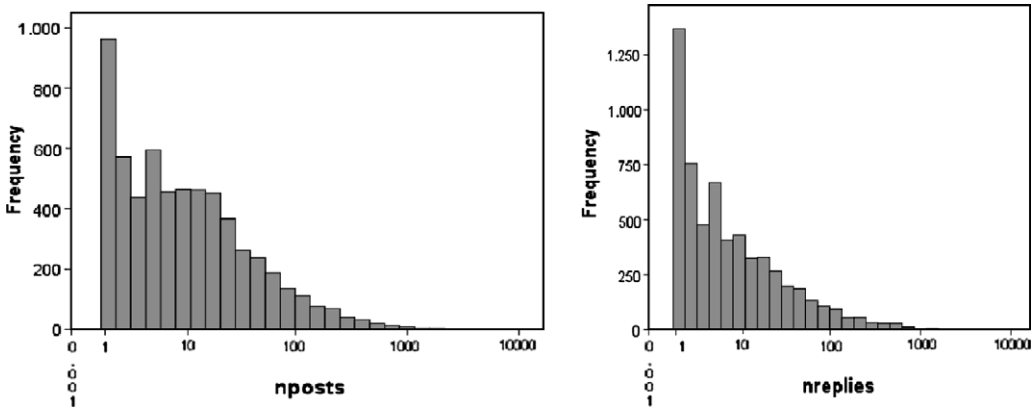

Fig. 4. Frequency distributions of posts and replies in the User list.

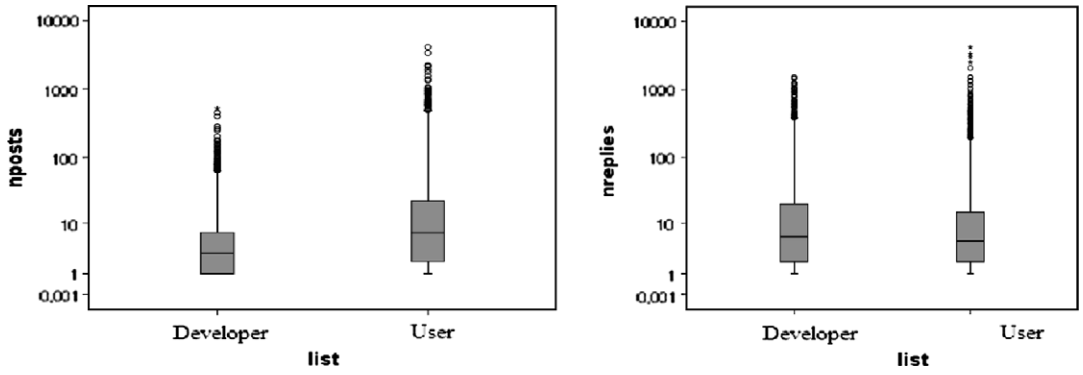

Fig. 5. Box-plots showing difference in posting and replying activities in the two lists.

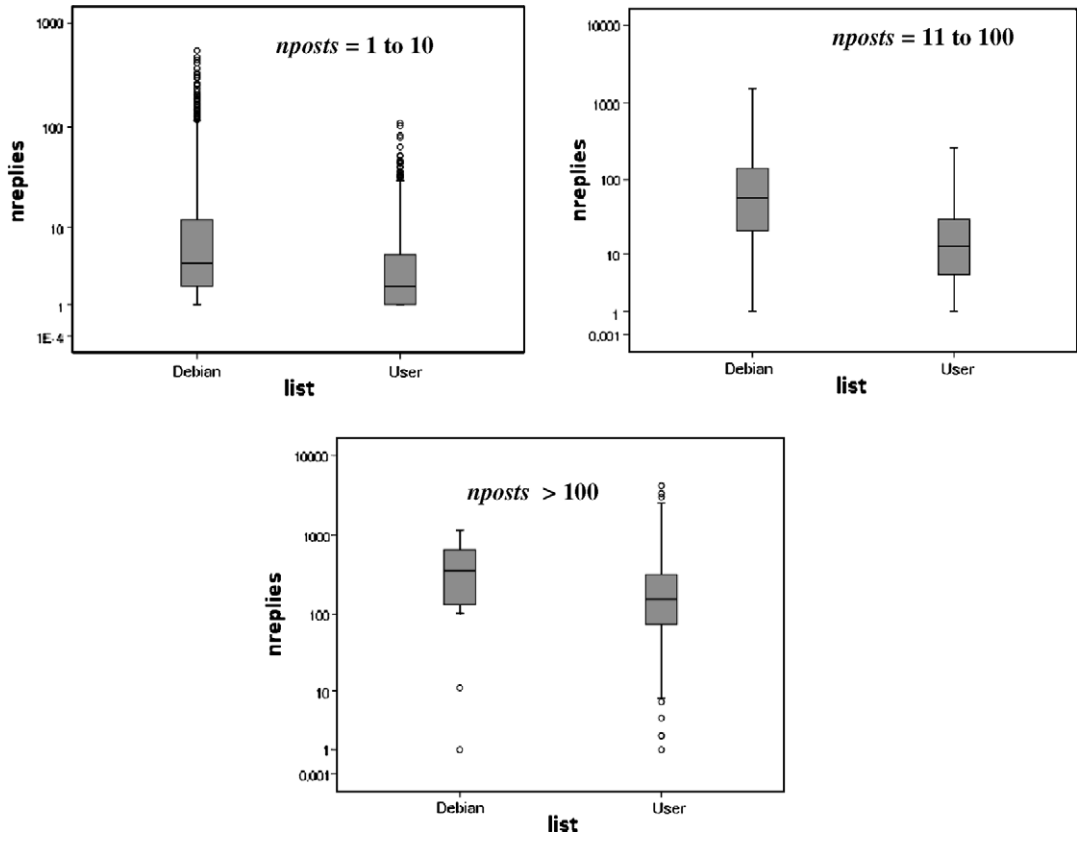

Fig. 6. Distribution of replies for groups of posts.

the two lists are shown in Table 4. While *nposts* are fairly highly correlated with *nreplies* in both lists, the correlation is stronger in the User list.

For the regression analysis, an exponential regression model for the Developer list explains only 44.6% ($r^2 = 0.446$) of the variability of *nreplies*, but this is statistically significant.

$$\ln(nreplies) = 0.892 + 0.933 * \ln(nposts)$$

or

$$nreplies = 2.44 * (nposts)^{0.933}$$

There are many outliers that make the regression model unreliable. However, it is useful to figure out the nature of their relation as shown in Fig. 8.

Regarding the User list, the Pearson correlation of the logarithms is 0.768, i.e. smaller than that of the original variables. For this case, it is not easy to derive a model.

$$nreplies = -2.333 + 0.928 * nposts$$

The linear model explains 85.7% ($r^2 = 0.857$) of the variability. However, residual analysis reveals several problems of reliability due to the large number of outliers. An alternative model is the exponential

$$\ln(nreplies) = 0.148 + 0.769 * \ln(nposts)$$

or

$$nreplies = 1.160 * (nposts)^{0.769}$$

Please cite this article in press as: Sowe, S.K. et al., Understanding knowledge sharing activities in free/open ..., J. Syst. Software (2007), doi:10.1016/j.jss.2007.03.086

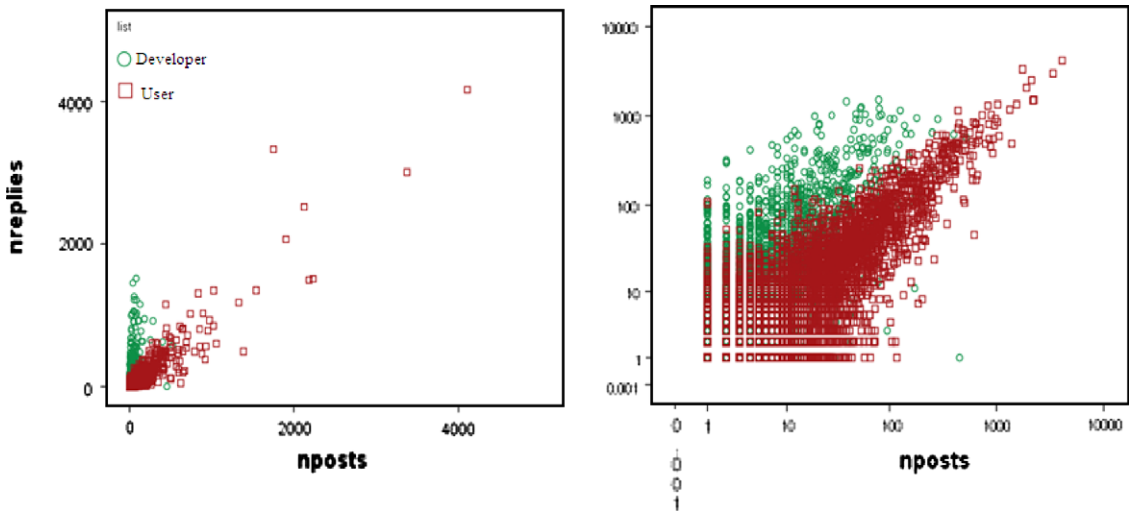

Fig. 7. Posts and replies superimposed in the two lists.

Table 4
Non-parametric correlations between posts and replies in the two lists

			Developer list		User list	
			ln(nposts)	ln(nposts)	ln(nposts)	ln(nposts)
Kendall's tau_b	Nposts	Correlation coefficient	1.000	.475[a]	1.000	.550[a]
		Sig. (2-tailed)	–	.000	–	.000
		N	3735	3735	5970	5970
	Nreplies	Correlation coefficient	.475[a]	1.000	.550[a]	1.000
		Sig. (2-tailed)	.000	–	.000	–
		N	3735	3735	5970	5970
Spearman's rho	Nposts	Correlation coefficient	1.000	.608[a]	1.000	.699[a]
		Sig. (2-tailed)	–	.000	–	.000
		N	3735	3735	5970	5970
	Nreplies	Correlation coefficient	.608[a]	1.000	.699[a]	1.000
		Sig. (2-tailed)	.000	–	.000	–
		N	3735	3735	5970	5970

[a] Correlation is significant at the 0.01 level (2-tailed).

This model explains only 59.0% ($r^2 = 0.590$) of the variability, but performs somewhat better in residual analysis as shown in Fig. 9.

5.3. Knowledge sharing in self-organizing communities

The distribution of the number of posts and replies in both lists shows that there is a "long tail" of posts and replies from the participants. A power law distribution of effort is expected in any self-organizing, self-reinforcing system such as F/OSS communities. The law has been found to hold for developer communities at Sourceforge – number of developers on a project and total number of projects-joined by a developer (Xu et al., 2005), F/OSS project sizes measured as the number of developers or lines of code (Koch, 2004; Hunt and Johnson, 2002), the distribution of knowledge brokers in F/OSS mailing lists (Sowe et al., 2006). For detail discussion of scale free networks and the application of power law on various aspects of our lives, see Barbasi and Bonabeau (2003) and Barbasi (2002). A power-law distribution is one in which frequent occurrences of an event (many posters or replies) are rarely observed, whereas few incidences of the same event (few posters or few replies) are very common. The validity of the power-law was tested on the posts and replies in both lists. First we rank the participants according to their postings (or replies) to a list (i.e. the one with the most postings takes rank one, the second takes rank two, etc.) and then we check whether these ranks have a linear relationship with the absolute number of postings (or replies) in a log–log scale. That means that if a participant has rank r and a number of postings (or replies) to a list is denoted by N, then the linear relation is of the form

$$N = cr^b \iff \log(N) = \log(c) + b\log(r);$$

where c and b are normalized constants

As shown in Fig. 10a and b, the linear correlation between r and N is strong ($\approx$1). The axes are in logarithmic scale. For each of the graphs the coefficient of determination (R^2), which is the proportion of variability in either the posts or replies, is also given.

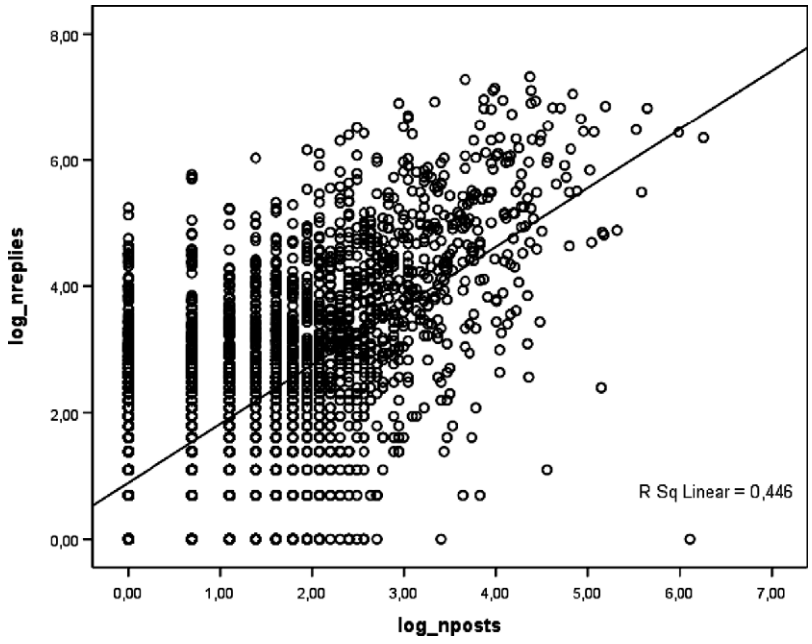

Fig. 8. Linear regression model for Developer list showing less than half of the variation in replies.

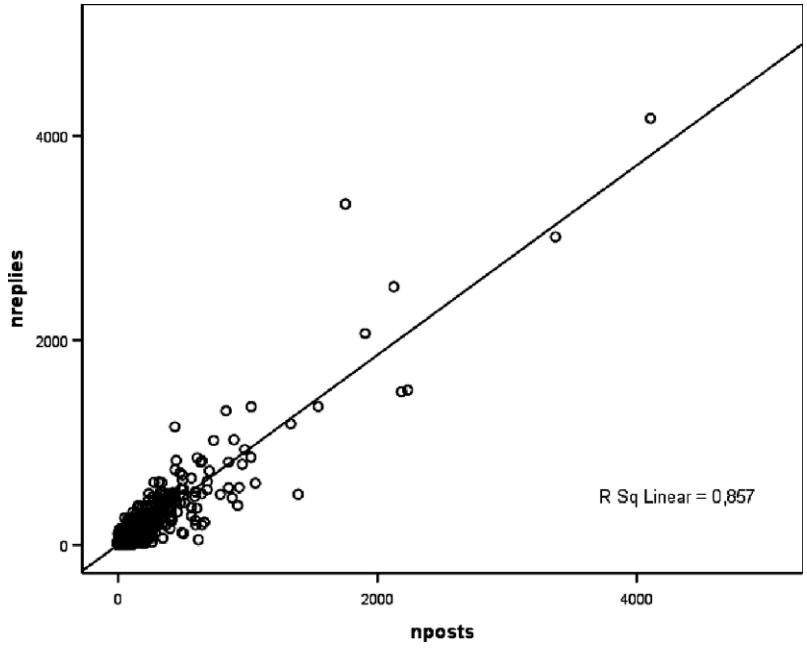

Fig. 9. Exponential regression model for User list showing more than half of the variation in replies.

However, what we found was that the linear relation is not satisfactory. Alternatively, we used a polynomial model of third order, i.e. a cubic relation of the type

$$\log N = b0 + b1 * \log r + b2 * (\log r)^2 + b3 * (\log r)^3$$

The graphs in Fig. 11 shows a perfect fit of the cubic polynomial. The coefficient of determination (r^2) = 1 in almost all cases. We call this type of distribution "cubic fractal distribution", since these distributions have some fractal (self-similarity) properties.

Fractals have been widely used to study complex relationships and were first introduced by Mandelbrot (1967), who described a power law distribution as one of fractal derivatives. The study of fractals is beyond the scope of this paper. In a recent study, Wu and Holt (2006) investigated the structural evolution of software systems in terms of the fractal distribution of source files. They observed that open source systems (OpenSSH, PostgreSQL and Linux Kernel) evolved through 'punctuated equilibrium' – an alternation between long periods of small

Please cite this article in press as: Sowe, S.K. et al., Understanding knowledge sharing activities in free/open ..., J. Syst. Software (2007), doi:10.1016/j.jss.2007.03.086

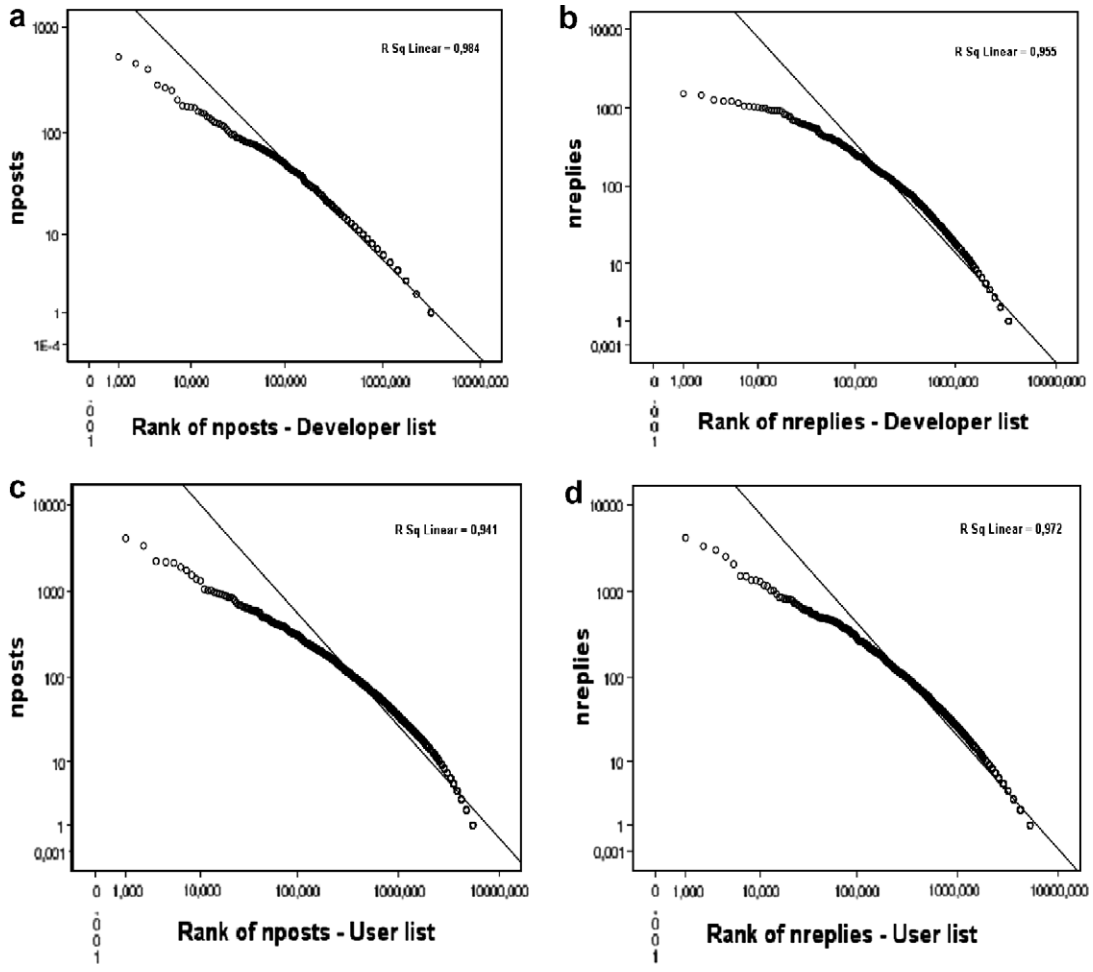

Fig. 10. Power-law distribution showing linear relation between participants' posts and replies and their ranks in the two lists.

changes and short periods of large changes. We also suspect that our data shows a cubic fractal distribution because the mailing lists seem to have a peak period when knowledge sharing is very intensive, followed by a period of 'silence' when knowledge sharing is less intensive.

6. Summary

In this paper we have discussed the knowledge sharing activities of individuals in two different kinds of lists (Developer and User) from the Debian project. The results and discussions we presented are meant to answer our research questions and can be summarized as follows.

Q1. *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, as Mockus et al. (2002) discovered in the developer mailing list of the Apache project, postings in the developer lists 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. For the User list, except where participants take the advice of Raymond and Moen (2006) on 'How To Ask Questions The Smart Way', not all the posts may be interesting to the participants.

Q2. *Is there a trend in the way individuals post and/or reply to questions in Developer and User lists?* We considered individuals who externalized posts and replies to the lists. In each list we had 'star contributors' who contributed most of the posts and replies. Furthermore, we grouped the posts into small, medium and large and found that participants with medium (11–100) and large (>100) posts contributed most of the replies. We found that whilst the User list participants contributed few posts and few replies, participants in the Developer list contributed many posts and many replies.

Q3. *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,

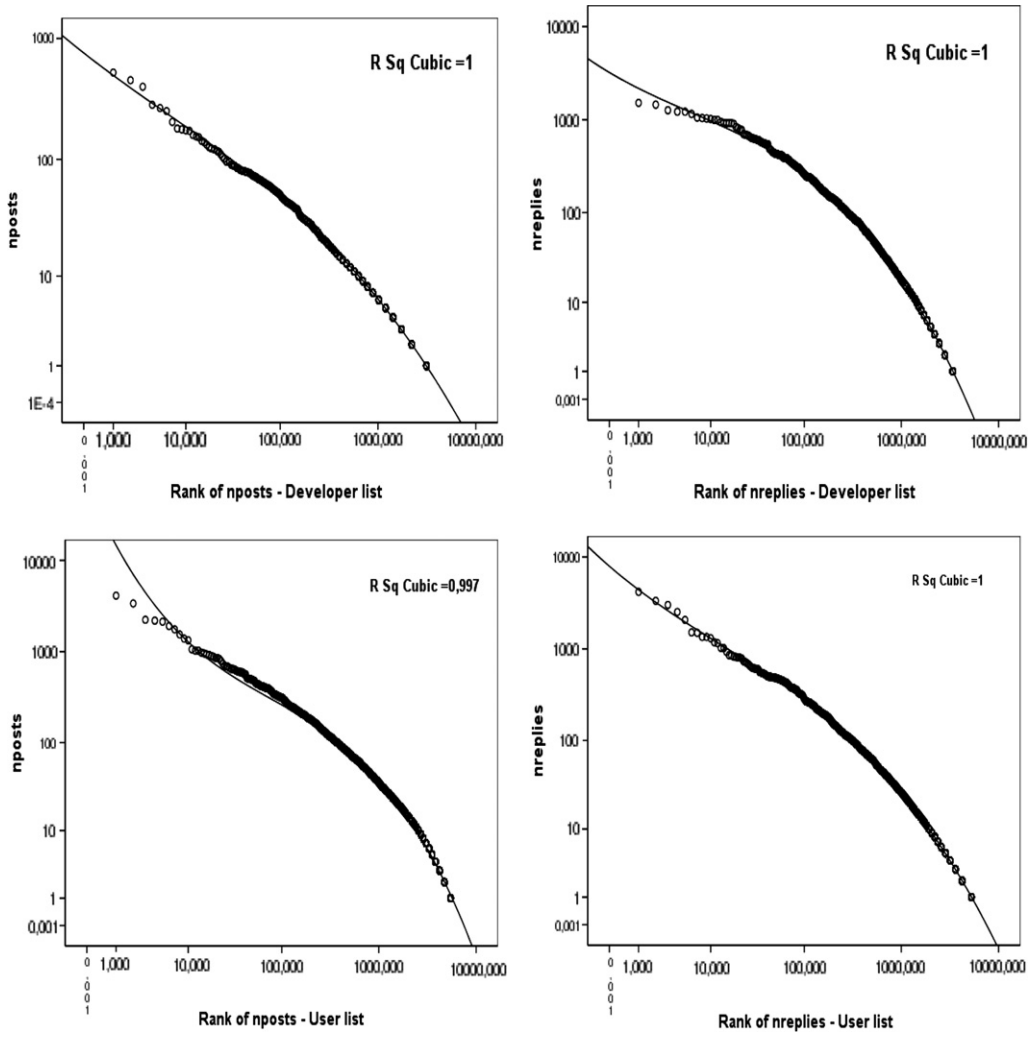

Fig. 11. "Cubic fractal distribution" or Cubic polynomial distribution of posts and replies in the two lists.

replying likewise increases. However, the correlation was stronger for the User list. Spearman's *rho* (ρ) was 0.608 for the Developer list and 0.699 for the User list. This means that, for the User list more questions generate more answers. On relating the posts with the replies, the exponential regression models explained the variability on the replies much better than the linear models.

Q4. *How can the knowledge sharing activities of Developer and User mailing lists participants be explained in terms of the self-organizing structure of F/OSS communities?* Communities formed around mailing lists are special kinds of self-organizing communities and knowledge sharing is an integral part of such communities. These communities are characterized by:

- Hierarchical structures and clear labour division (Valverde et al., 2006): In each of the mailing lists, individuals tend to adopt specialized and unassigned roles, as either knowledge seekers or knowledge providers or both, determined by the types of emails exchanged (posts or replies). When we rank the individuals according to their postings or replies, a hierarchical structure emerged in which some individuals dominated posting and replying activities. However, the dominant role of an individual is not always stable.
- We removed the top-rank individuals, Valverde et al. (2006) called 'α-individuals', in each list and studied their contributions in respect to their roles as either posters or repliers. In our case, these individuals are the "Star contributors". Dominant repliers tend to be long-term members of the community. This is expected because knowledge providers are assumed to be well experienced and know the software well. On the contrary, we expect dominant posters to be 'young' in the community, less experienced, and ask a lot of questions. However, we found that the star contributors first appeared much earlier in the Developer list. The dominant poster first appeared on November 3rd, 2000 and the dominant replier first appeared on January 21st, 2000. For the User list the dominant replier first appeared on December 12th, 2001, and the dominant poster first appeared on March 3rd,

2005. Thus, while our expectation holds for the User list, the situation was different for the Developer list.
- The distribution of the posts and replies in both lists follow a power-law distribution. However, the distribution of these activities could best be explained using what we called *'fractal cubic distributions'*.

6.1. Validity threats

While Debian might be said to be representative of a successful F/OSS project, we have selected only two lists to analyze knowledge sharing activities in the project. Thus, there is danger in generalizing the results to other lists in other projects where the nature of knowledge sharing may be different. However, as Mockus et al. (2002) found out in their study of the Apache web server, the analysis of a single case can provide important insights and ground for future research in this area.

The data cleaning procedure we discussed in Section 4.3 resulted in the removal from our data of about 50% of the individuals from the User List and about 66% of the individuals from the Developer List. These are individuals who only posted emails and made no replies and vise versa. The removal was necessary in order to account for knowledge sharing between the participants. This means that individuals who only posted replies to questions asked in the lists were excluded in our study. We posit that these excluded individuals may play an important role in the knowledge sharing dynamics of the lists because by replying to questions, they help participants in the project use and benefit from the software.

6.2. Future work

Understanding knowledge sharing in F/OSS projects by analyzing email exchanges in mailing lists is an open research area that requires more attention. While we have attempted to answer few questions which may help us understand knowledge sharing activities in F/OSS projects, we have also opened a number of avenues for future research. Some of these are:

- We plan to study five more developer and user lists in the Gnome, FreeBSD, KDE, Apache, and Firefox projects to see if the pattern of knowledge sharing we found in the lists just studied can be generalized.
- The distribution of the posts and replies exhibits some self-similarities or fractals, suggesting that 'something' complex and interesting is going on in knowledge sharing in F/OSS projects that is worth further investigation.
- As a measure to add value to the empirical analysis we presented here, we are currently conducting an online survey with the lists participants asking them, among other questions, to identify their association (Developer, User), role (poster or replier), how they view their contribution to the Debian project, etc.

We also conjecture some questions, answers to which may add to our further understanding of the nature of knowledge sharing activities in F/OSS.

- If a participant posts a question to a list what is the probability that he/she will receive a reply?
- For individuals who only reply to questions others post in the list and may never ask questions, what is the motivation for this activity?
- What is the probability that he/she will receive more than one reply?'' Answers to this question would help mailing lists participants decide whether to enter one project/list or another.
- Does the intensity of knowledge sharing vary from one kind of list to another, i.e. from Developer to User list?

Our exposition in this paper, especially the theoretical foundation, data extraction and cleaning methodology, and analysis may serve as an important structure for future researchers in this area. For persons interested in replicating this study, text files of the clean data, individuals names removed for anonymity, used in this study can be freely obtained from our website http://sweng.csd.auth.gr/~sksowe/JSS/ or by contacting the authors.

7. Conclusion

This paper has investigated knowledge sharing activities in F/OSS projects mailing lists using Debian as a case study. A knowledge sharing model was used to discuss knowledge sharing dynamics and show how list participants externalize knowledge into and internalize knowledge from a project's mailing list. We discussed a methodology to identify individuals and extract posts and replies from mailing lists archives. Metrics were used to analyze the knowledge sharing activities of the lists participants on various activity measures. In the analysis we discussed; the posting and replying activities of the participants and how much knowledge is externalized and internalized into each lists, the trend in knowledge sharing in the lists, the correlation between posting and replying activities, and the self-organizing nature of the individual's knowledge sharing activities. The measures by which we have analyzed knowledge sharing activities will not only help us understand and learn from the way knowledge is shared and created in F/OSS projects, but provide researchers in the field the initial stage for the generation of hypothesis and research questions.

Acknowledgements

We are grateful to Martin Michlmayr of the Debian project for providing us the data used in this paper and

for his comments on the early draft. We are also grateful to Dr. Fotis Kokkoras of the Department of informatics, Aristotle University for his advise on the data extraction methodology.

References

Atreyi, K., Bernard C.Y.T., 2004. A review of metrics for knowledge management systems and knowledge management initiatives. In: Proceedings of the 37th Annual Hawaii International Conference on System Sciences, p. 80238a.

Barahona, J., Robles, G., Perez, M., Merino, L., Olivera, V., Barbero, E., Quiros, P., 2004. Analysing the anatomy of GNU/Linux distributions: methodology and case studies (Red Hat and Debian). In: Koch, S. (Ed.), Free/Open Source Software Development. Idea Group Inc., pp. 27–58.

Barbasi, A., 2002. LINKED, The New Science of Networks, Cambridge, MA.

Barbasi, A., Bonabeau, E., 2003. Scale-free networks. Scientific America 288 (5), 50–59.

Conklin, M., 2006. Beyond low-hanging fruit: seeking the next generation in FLOSS data mining. In: Damiani, E., Fitzerald, B., Scacchi, W., Scott, M., Succi, G. (Eds.), In: Open Source Systems. International Federation for Information Processing, vol. 203. Springer, Boston, p. 48.

Davenport, H.T., Prusak, L., 2000. Working Knowledge. How Organizations Manage What They Know. Harvard Business School Press, pp. 81–83.

Debian Mailing Lists. Available: http://lists.debian.org/, accessed 12/6/2006.

Fitzerald, B., 2004. A critical look at open source. Computer 37 (7), 92–94.

German, D.M., 2004. Using software trails to reconstruct the evolution of software. Journal of Software Maintenance and Evolution: Research and Practice 16 (6), 367–384.

German, D.M., Mockus, A., 2003. Automating the Measurement of Open Source Projects, In: ICSE '03 Workshop on Open Source Software Engineering, Portland, Oregon, May 3–10.

Ghosh, R.A., 2004. Clustering and dependencies in Free/Open Source Software Development: methodology and tools. Available: http://www.firstmonday.dk/issues/issue8_4/ghosh/, Accessed; 12/08/2006.

Gloor, P.A., Laubacher, R., Dynes, S.B., Zhao, Y., 2003. Visualization of communication patterns in collaborative innovation networks analysis of some W3C working groups. Presented at ACM CKIM International Conference on Information and Knowledge Management, New Orleans, November 3–8, Knowledge Management Session 1, pp. 56–60.

Hahsler, H., Koch, S., 2005. 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), January 03–06, Big Island, Hawaii, p. 197b.

Healy, K., Schussman, A., 2003. The ecology of open-source software development. Available: http://www.kieranhealy.org/files/drafts/oss-activity.pdf, accessed 6/11/2005.

Hertel, G., Niedner, S., Herrmann, S., 2003. Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux Kernel. http://opensource.mit.edu/papers/rp-hertelniednerherrmann.pdf, accessed 02/06/2004.

Hunt, F., Johnson, P., 2002. On the Pareto distribution of Sourceforge projects. In: Proceedings of Open Source Software Development Workshop, Newcastle, UK, pp. 122–129.

Internet Message Format (RFC) 2822, 2001. Network Working Group. Available: http://rfc.giga.net.tw/rfc2822, accessed 12/12/2006.

Kim, E., 2003. An introduction to open source communities. Blue Oxen Associates. Available: http://www.blueoxen.org/research/00007/BOA-00007.pdf, last accessed 19/01/2007.

Koch, S., 2004. Profiling an open source project ecology and its programmers. Electronic Markets 14 (2), 77–88.

Koch, S. (Ed.), 2004. Free/Open Source Software Development. Idea Group Inc., pp. vii–viii.

Koch, S., Schneider, G., 2002. Effort, cooperation and coordination in an open source software project: Gnome. Information Systems Journal 12 (1), 27–42.

Koh, J., Kim, Y., 2004. Knowledge sharing in virtual communities: an e-business perspective. Expert Systems with Applications 26, 155–166.

Krishnamurthy, S., 2002. Cave or community? An empirical examination of 100 mature open source projects. Firstmonday 7 (6). Available: http://firstmonday.org/issues/issue7_6/krishnamurthy/index.h.

Krogh, G., Spaeth, S., Lakhani, K., 2003. Community, joining, and specialisation in open source software innovation: a case study. Research Policy 32, 1217–1241.

Lakhani, K., Hippel, E., 2003. How open source software works: "free" user-to-user assistance. Research Policy 32, 923–943.

Lanzara, G.F., Morner, M., 2003. The Knowledge Ecology of Open-source Software Projects, European Group of Organizational Studies (EGOS Colloquium), Copenhagen.

Mandelbrot, B., 1967. How long is the coast of Britain? Statistical self-similarity and fractional dimension. Science 156, 636–638.

Michlmayr, M., 2004. Managing volunteer activity in free software projects. In: Proceedings of the 2004 USENIX Annual Technical Conference, Freenix Track, pp. 93–102.

Mockus, A., Fielding, R., Herbsleb, J.A., 2002. Two case studies of open source software development: Apache and Mozilla. ACM Transactions on Software Engineering and Methodology 11 (3), 1–38.

Nichols, D., Twidale, M., 2003. The usability of open source software. Firstmonday 8 (1).

Nonaka, I., Takeuchi, H., 1995. The Knowledge Creating Company. Oxford University Press.

Raymond, E.S., 1999. The Cathedral and the Bazaar. Firstmonday 3 (3). Available: http://www.firstmonday.org/issues/issue3_3/raymond/.

Raymond, E.S., Moen, R., 2006. How To Ask Questions The Smart Way. Revision 3.2. http://www.catb.org/~esr/faqs/smart-questions.html, last accessed; 18/11/2006.

Scacchi, W., 2006. Understanding Free/Open Source Software Development processes. Software Process Improvement and Practice 11 (2), 95–105.

Schofield, A., Cooper, G.S., 2006. Participation in F/OSS communities. An empirical study of community members' perceptions. In: Damiani, E., Fitzerald, B., Scacchi, W., Scott, M., Succi, G. (Eds.), . In: Open Source Systems. International Federation for Information Processing, vol. 203. Springer, Boston, pp. 221–231.

Sowe, S.K., Karoulis, A., Stamelos, I., Bleris, G.L., 2004. Free-open source learning community and web-based technologies. IEEE Learning Technology Newsletter 6 (1), 26–29.

Sowe, S.K., Karoulis, A., Stamelos, I., 2005. A constructivist view of knowledge management in open source virtual communities. In: Figueiredo, D.A., Paula, A. (Eds.), Managing Learning in Virtual Settings: The Role of Context. Idea Group Inc., pp. 290–308.

Sowe, S.K., Stamelos, I., Angelis, L., 2006. Identifying knowledge brokers that yield software engineering knowledge in OSS projects. Information and Software Technology 48, 1025–1033.

Sowe, S.K., Stamelos, I., Deligiannis, I., 2006. A framework for teaching software testing using F/OSS methodology. In: Damiani, E., Fitzerald, B., Scacchi, W., Scott, M., Succi, G. (Eds.), . In: Open Source Systems. International Federation for Information Processing, vol. 203. Springer, Boston, pp. 261–266.

Sowe, S.K., Angelis, L., Stamelos, I., Manolopoulos, I., 2007. Using Repository of Repositories (RoRs) to study the growth of F/OSS projects: a meta-analysis research approach. In: Proceedings of the Third International Conference on Open Source Systems, June 11–14 2007, Limerick, Ireland.

Timothy, L., Elliott, S., Singer, J., 2005. Studying software engineers: data collection techniques for software field studies. Empirical Software Engineering 10, 311–342.

Valverde, S., Theraulaz, G., Gautrais, J., Fourcassie, V., Sole, R.V., 2006. Self-organization patterns in wasp and open source communities. IEEE Intelligent Systems 21 (2), 36–40.

von Krogh, G., Spaeth, S., Haefliger, S., 2005. Knowledge reuse in open source software: an exploratory study of 15 open source projects. In: Proceedings of the 38th Hawaii International Conference on System Sciences, (IEEE, HICSS '05-Track 7), January 03–06, Big Island, Hawaii, p. 198b.

Wu, J., Holt, R., 2006. Seeking empirical evidence for self-organized criticality in open source software evolution. Available: http://www.cs.uwaterloo.ca/research/tr/2006/CS-2006-14.pdf, last accessed 02/12/2006.

Xu, J., Gao, Y., Christley, S., Madey, S., 2005. A topological analysis of the open source software development community. In: IEEE Proceedings of the 38th Hawaii International Conference on System Sciences (IEEE, HICSS '05-Track 7), January 03–06, Big Island, Hawaii, p. 198a.

Ye, Y., Kishida, K., 2003. Towards an Understanding of the motivation of open source software developers. In: Proceedings of the 25th International Conference on Software Engineering, Portland, Oregon, May 3–10, pp. 419–429.

Ye, Y., Yamamoto, Y., Kishida, K., 2004. Dynamic community: a new conceptual framework for supporting knowledge collaboration in software development. In: Proceedings of the Asia Pacific Software Engineering Conference, pp. 472–481.

Zack, H.M., 1998. Developing a knowledge strategy. Sloan Management Review 41 (3), 125–145.

Zeldin, T., 1999. Creating a knowledge sharing culture. Knowledge Management Magazine 2 (5), 213–238.

Please cite this article in press as: Sowe, S.K. et al., Understanding knowledge sharing activities in free/open ..., J. Syst. Software (2007), doi:10.1016/j.jss.2007.03.086