Available online at www.sciencedirect.com
ScienceDirect
Information and Software Technology 48 (2006) 1025–1033
INFORMATION AND SOFTWARE TECHNOLOGY
www.elsevier.com/locate/infsof

Identifying knowledge brokers that yield software engineering knowledge in OSS projects

Sulayman Sowe *, Ioannis Stamelos, Lefteris Angelis

Department of Informatics, Aristotle University of Thessaloniki, 54124 Thessaloniki, Greece

Received 17 June 2005; received in revised form 19 December 2005; accepted 23 December 2005
Available online 3 March 2006

Abstract

Much research on open source software development concentrates on developer lists and other software repositories to investigate what motivates professional software developers to participate in open source software projects. Little attention has been paid to individuals who spend valuable time in lists helping participants on some mundane yet vital project activities. Using three Debian lists as a case study we investigate the impact of knowledge brokers and their associated activities in open source projects. Social network analysis was used to visualize how participants are affiliated with the lists. The network topology reveals substantial community participation. The consequence of collaborating in mundane activities for the success of open source software projects is discussed. The direct beneficiaries of this research are in the identification of knowledge experts in open source software projects.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Open source software projects; Debian; Mailing lists; Mundane activities; Social networks; Visualization

1. Introduction

Open source software (OSS) development not only exemplifies a viable software development approach, but also a model for the creation of self-learning and self-organizing communities. Enable by the Internet, geographically distributed individuals voluntarily contribute to a project by means of the Bazaar model [29]. Extensive peer collaboration allow project participants to write code, debug, test, and integrate software. Numerous research findings established that OSS participants are motivated by a combination of intrinsic and extrinsic motives [8,16,20,23]. The most typical characteristic of the OSS development process is that developers are themselves users of the software. And thus, are better positioned to fix problems or bugs they encounter in the software. Problems along with the necessary steps taken to correct them are then reported back to the community for all to benefit. As Keith [18] expressed in his 'Ten Worst Engineering Pitfalls', the technical savvyness of OSS is perhaps one of its pitfall. Users of OSS are expected to possess enough technical knowledge and know how to configure the software before using it.

At the beginning when Linux just came into the scene in the early 1990s, volunteers had many choices and it was not much of a problem for OSS projects to attract massive and talented developers *cum* users, testers, debuggers, etc. The concept was new, OSS was built and projects and communities sprung in large numbers. Success stories are registered in the area of operating systems (Linux), Web Services (Apache), scripting programming languages (Python, Perl), database applications (MySQL, PostgreSQl), etc. The boon has changed the ecology and dynamics of open source software. Anecdotal evidence reported a new wave of Linux 'migrating users' who are young, have significant user experience, ask a loft of questions, have an interest in learning and using OSS [1]. Furthermore, a large numbers of non-technical *end-users* are participating in OSS projects [28]. They get involved in mundane activities such as suggesting new features, testing, writing documentation, reporting bugs, software localization/

* Corresponding author. Tel.: +30 2310 991927; fax: +30 2310 998419.
E-mail address: sksowe@csd.auth.gr (S. Sowe).

0950-5849/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.infsof.2005.12.019

internalization [8]. In their study of the Apache field support system, Lakhani and von Hippel [21] argued that a complete OSS project requires the execution of mundane activities such as field support. Bonaccorsi and Rosi [5] added that these activities are fundamental for the adoption of Open source software. We posit that mundane activities are vital for the success of OSS projects. Furthermore, the interaction between knowledge seekers and knowledge providers, and between the developers and users of the software, play a major role in the diffusion and improvement of the software. Yet, little rigorous research has been conducted in this area.

In this paper, we aim at developing a better understanding of these activities and interactions and their importance for the success of OSS projects. We discuss one group of OSS project participants—*knowledge brokers*—who bridge the gap between expert software developers and user communities. Knowledge brokers not only help individuals manage and extract valued software knowledge from software repositories, but also help OSS projects to engage in a discourse and co-learning experience with their user communities [30]. Software repositories explored in this paper are mailing lists, where community activities are not confined to software development or coding alone. Apart from being the main communication channel in most projects, lists enable knowledge brokerage by linking project participants, enriching their interaction so that both are able to benefit each other and the project. But, the very nature of email threads makes it difficult to see who is communicating with whom, how much discussion is going on, and who the major contributors are. Identifying knowledge brokers within the array of lists participants is an intricate task. Especially in long threads often spanning of several email exchanges and when email messages are archived. We address this problem by presenting a methodology to extract data from three non-developer mailing lists from the Debian project. We counted the number of posts to the lists and identified knowledge brokers who posted software information and made replies to questions others posted to the lists. Social networks visualization technique was used to visualize how the knowledge brokers interacted with other participants in the lists.

2. Background and related work

Much research has been done on how software developers work in OSS projects and what motivates them to take part in coding activities [7,16,20,26]. Other studies give insight into the nature of community participation in OSS projects mailing lists [10,19,20,26]. Our study builds upon these researches, focusing specifically on the activities of knowledge brokers in non-developer mailing lists. With the exception of few studies (e.g. [14,21,22]), there is less focus on knowledge sharing and mundane activities in OSS projects. We conjecture that non-developer lists are important knowledge repositories and they can be data mined in order to discover patterns of knowledge sharing activities. Structurally, lists act as proxies for every individual subscribed to them [13], making it easy for project participants to find out who the experts are in a given area. Once a list is identified, an individual may subscribe and post his questions. An expert in that area will reply or redirect him to an appropriate link or tool.

Social networks analysis provides much insight on how software developers collaborated in the software development process across projects. In [24,31], software developers form nodes on a network (graph) and links or edges exist between the nodes if developers participate on the same project. In these studies, identification of clusters exposed connected groups of software developers across OSS projects hosted at the famous OSS portals (`sourceforge.net` and `freshmeat.net`). The networks revealed a small group of highly prolific software developers or 'linchpins'. In our mailing list network, we also found out that participants in one list are connected to participants in another list in the Debian project. The presence of knowledge brokers in our visualized network is akin to the discovery of linchpin developers in the above studies. A number of studies have employed social network visualizations to visualize communication flow among individuals by email logs [12], to study interaction among IRC channels users [27], and to visualize the relationships among modules in the Apache project [2].

3. Methodology

Like source code repositories, open source projects mailing list data is widely available and easy to extract. This provides an excellent infrastructure to study community participation in OSS projects [10,26]. Our study utilized data from the Debian project's lists archives. Debian was selected because the project provides opportunities for researchers (e.g. [25]) to observe open source software community participation. Fig. 1 shows a flowchart of the methodology we used in this study.

3.1. Data collection

The Debian project hosts over 100 lists on all aspects related to the project. We selected three lists from the Debian lists server (`lists.debian.org`). The main selection criteria include: list lifespan, activity, and openness. The lists are two-way and unmoderated. Participants can subscribe to and freely post questions and get replies. Contributions to a list are made in the form; debian-listname@lists.debian.org. The following *listnames* are analyzed in our study:

- **Kde**: discussions on issues related to KDE in Debian.
- **Mentors**: dedicated to novice Debian developers and new package maintainers. Participants can seek help with packaging and other developer related issues here. This is often the entry point to many first-time Debian participants.

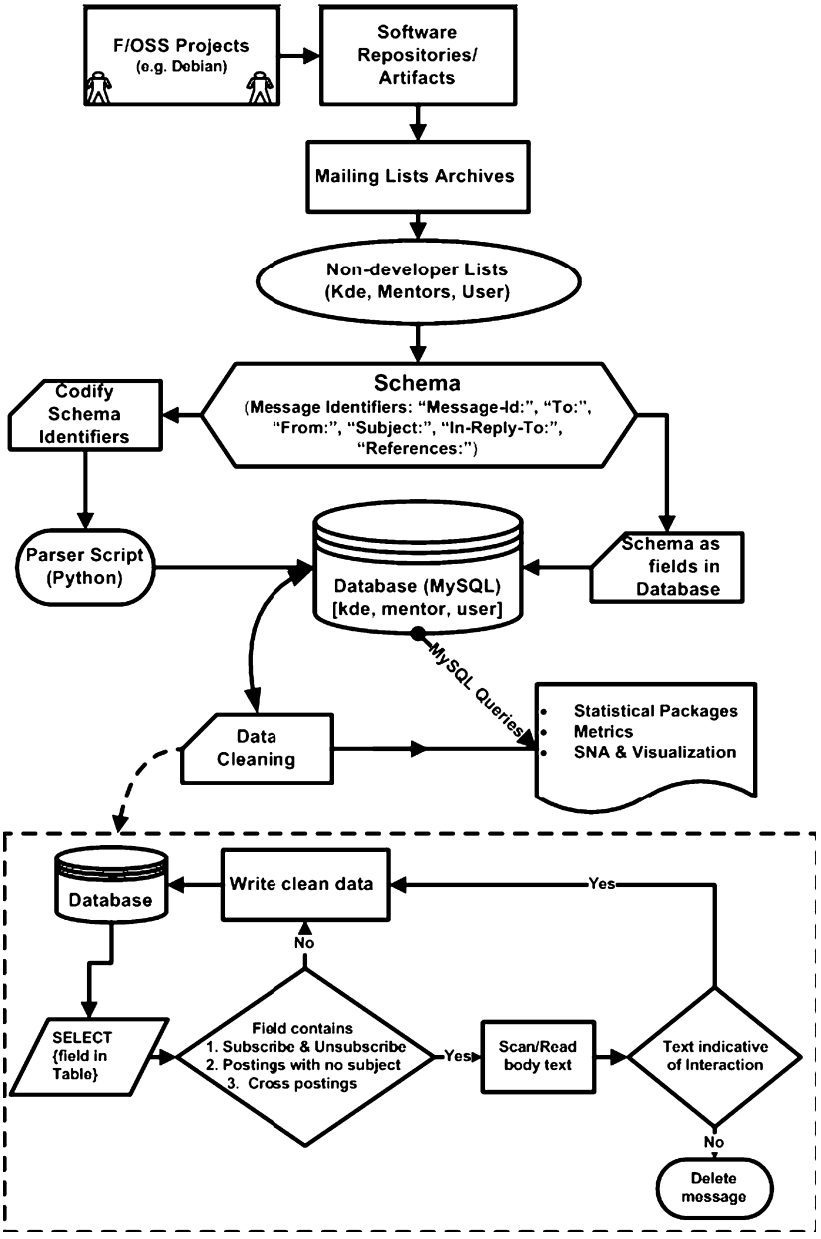

Fig. 1. Methodological outline.

- **User**: meant for Debian users who speak English. There are other Debian-user lists in other languages (e.g. French, German, Spanish, etc.). The list is a 'high-volume mailing list' where almost all issues relating to the Debian project are discussed.

The three lists were created at different times and are still active. The oldest among them, the User list, was created in 1995. The Mentors and Kde lists were created in 1998 and 2000, respectively. In order to ensure uniformity in our data, the collection period was from January 2001 to September 2004. However, in selecting a study period, we suffered from the 'right censoring effects' [20]. Some of the replies we counted are replies to messages posted prior to the time we started our data extraction. Some posts we included in our data may have replies beyond our coverage and were not counted.

We obtained archived mbox files of the three lists. Each file is a single text file containing 1 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. A coding schema was developed to extract the following message identifiers and map them as fields in a database:

- 'Message-Id:' uniquely identifies a message posted by a participant.
- 'To:' destination of the message.
- 'From:' identifies the origin (poster) of a message.

- 'Subject:' heading identifying the subject of the posted message.
- 'In-Reply-To:' identifies a reply to a given posting.
- 'References:' if any, points to the origin of the current message.
- 'Body:' plain text containing the email message.

3.2. Identification of individuals

From the lists two types of individuals could be identified, *posters* and *repliers*. 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 'From' filed 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 'From' 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 [11]. Where the same poster used different emails, we counted him as one. Using the schema, we had three means of identifying whether some is a replier. Where the person sends a message which has 'Re:' in the 'Subject' field, where the 'In-Reply-To' and 'References' fields corresponding to that person are not null.

For the identification of posters and repliers, we implemented a modified version of Zawinski's threading algorithm [32] and tested it on the Kde list. The test resulted in 14,178 messages with 'Re:' contained in the 'Subject' field. The 'In-Reply-To' field had 13,256 messages, and the 'References' field yielded 13,254 replies. Messages with 'Re:' in the 'Subject' field form the largest proportion, by a small magnitude. Thus, posted messages with 'Re:' in the 'Subject' field were chosen as the best means to identify repliers, and otherwise posters. An earlier study implemented a similar algorithm and computed some weighted measures% using Bayes' Theorem to conclude that the chance of a message being labeled as a reply using this method was 99% [17].

3.3. Data extraction

A Python script implementing the schema was used to extract data from the mbox files. For a given input list (Kde, Mentors, User), the script traversed each mbox to extract a record for each *Message-Id*(primary key). The output for each run was parsed into a MySQL database containing three tables, one for each list. SQL queries were used to extract information necessary for data analysis.

3.4. Data cleaning

For participants to interact they must exchange email messages. A participant posts a message to a list and may get a reply from another participant. Fig. 2 depicts this kind of interaction as a cycle where posters (knowledge seekers and/or knowledge providers) are continuously internalizing and externalizing knowledge into the mailing lists.

Some postings may not be answered immediately, but may be viewed as important knowledge externalized into the list from which others may benefit. Participants may also be involved in private email or off-list discussions that are not posted to the lists due to their private nature [30] (p. 302). In order to improve the quality of our data and account for participants' interaction, certain postings had to be removed. The data cleaning process involved removing messages in the following categories:

(1) **Subscribe and unsubscribe messages:** The three tables were queried for email messages with 'subscribe' and 'unsubscribe' in the 'Subject' field. Some messages in this category were found to contain information from which subsequent participants may benefit. Consider the example below where the subject header had 'unsubscribe me'.
   - **Post**: *Am having difficulty unsubscribing from this list. Can anyone help?*
   - **Reply**: *If you think another address might have been used, try saying unsubscribe* [somebody@somewhere.gm] *or whatever rather than unsubscribe.*

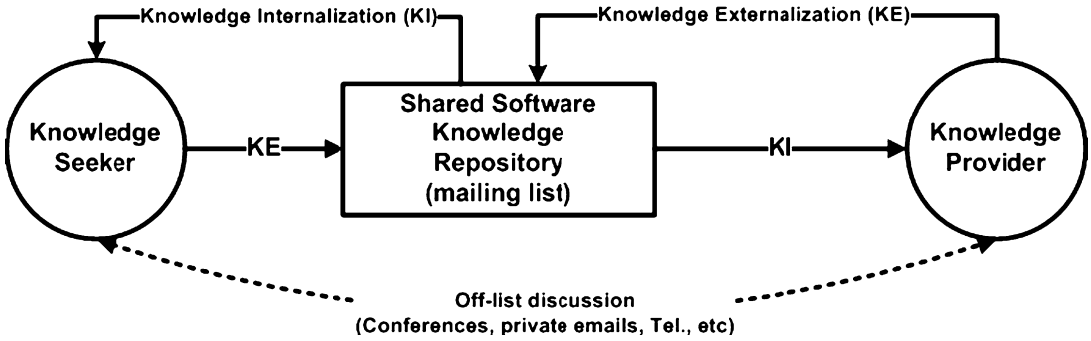

Fig. 2. Knowledge sharing in mailing lists.

- **Post**: *That worked! At least I got a message that said I was removed from the list. But who knows...I will see in a couple of minutes, I think. Thank you very much.*
- **Post**: *I hope it continues to work.:) Cc'ed to you directly, as assuming it worked you won't see the reply otherwise.*

With this kind of interaction, subsequent participants will have information on how to unsubscribe from the list. After inspecting the queries for genuine 'subscribed' and 'unsubscribed' messages, 301 (1.50%) messages were removed from the Kde, 96 (0.70%) from the Mentors list, and 2465 (1.10%) from the User list, respectively.

(2) **Postings with no subject:** The three tables were queried for 'empty' subject headers. However, after scanning through what was supposed to be 'empty' message bodies, were found text which contain genuine questions addressed by posters. For example:
- **Poster:** *Hello, I get the following errors when I try to compile an application using a structure I declared into an archive (.a file). The structure st_connection Data is properly defined in confighelper.h and confighelper.h is included in flatfilehelper.h. Any idea how to work around this?* [**Poster** listed error report]. *Thanks*

Further cleaning was carried out to ensure that the same sender did not repost the same message with a subject. After querying and confirming the 'Subject' field for 'empty' postings, 20 (0.1%) messages were removed from the Kde list, 27 (0.20%) from the Mentors list, and 269 (0.12%) from the User list, respectively.

(3) **Cross postings:** A poster sending the same information to all the lists. Some cross postings were not removed from our data. For example, a module or package maintainer may post software updates or patches to more than one list. The drawback in this is that our posters' count is larger by a certain magnitude, giving the impression that there are fewer replies to questions posters asked. Other crosspostings, where a participant asks the same question in all the lists, were removed. A python script was used to match the occurrence of the same string in the 'Subject' field in each of the three tables and output the results for inspection. Thirty-seven cross postings were identified, inspected, and removed.

4. Results and discussions

The total number of posts and replies obtained from each list after data cleaning is shown in Table 1. For each of the Kde, Mentors, and User lists we introduce the variables *Pkde, Pmentor, Puser* and *Rkde, Rmentor, Ruser* to represent the postings and replies made to the lists.

Table 1
Descriptive statistics of variables

	Pkde	Pmentor	Puser	Rkde	Rmentor	Ruser
N						
Valid	2906	2195	19,832	4061	2590	43,066
Missing	0	0	0	0	0	0
Mean	6.90	6.22	11.30	3.49	3.83	3.86
Median	2.00	2.00	2.00	2.00	2.00	2.00
Mode	1	1	1	1	1	1
SD	25.820	17.567	68.246	4.139	4.090	5.778
Range	746	414	4105	75	46	256
Minimum	1	1	1	1	1	1
Maximum	747	415	4106	76	47	257
Sum	20,053	13,653	224,053	14,180	9918	166,288

The descriptive statistics shows that 2906 individuals posted 20,053 email messages to the Kde list and 4061 generated 14,180 replies. In the Mentors list 2195 individuals posted 13,653 email messages and 2590 generated 9918 replies. For the User list, 19,832 individuals posted 224,053 email messages and 43,066 generated 166,288 replies. The STD deviations shows skewness in the distribution of postings and replies. A small number of participants dominated posting and replying activities in all the three lists. For example, in the Kde list one poster was responsible for about 37% of the posted email messages. The mean number of postings per person was highest for the User list (11.30) and lowest for the Mentors list (6.22). The mean number of replies per person is almost constant, with a mean value of approximately four.

For the reason that our focus is on the identification of knowledge brokers, we need a method to identify participants who generated and shared knowledge across the lists. That is, which knowledge providers rendered help in the lists. How much they contributed by positing answers to questions knowledge seekers asked. Knowledge seekers may also ask different questions in different lists.

4.1. Method to identify knowledge brokers

Because list participants are linked by virtue of the messages they exchange, we queried the three tables and found 136 participants. This cohort represents posters

Table 2
Descriptive statistics of the 136 participants

	Kde	Mentors	User
N			
Valid	136	136	136
Missing	0	0	0
Mean	18.15	19.21	75.66
Median	2.00	4.00	5.50
Mode	1	1	1
SD	60.246	49.597	367.657
Minimum	1.00	1.00	1.00
Maximum	557	415	4106
Sum	2468	2612	10290

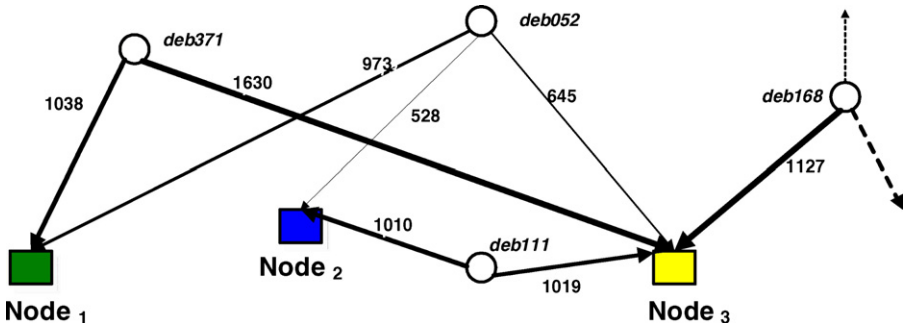

Fig. 3. Mailing list bipartite graph showing lists as nodes.

who either posted answers to questions knowledge seekers asked in the three lists or posted information to all the lists. Information posted to the lists includes product updates, patches, release information, etc. Table 2 shows summary statistics of the data for this cohort.

Like the posting and replying activities in the individual lists, the 136 cohort shows similar posting pattern. Certain individuals contributed most of the postings. For example, about 40% of the replies to questions participants asked in the User list was made by one person. Again, the mean number of postings per person was highest for the User list (75.66) but lowest for the Kde list (18.15), showing heterogeneity in knowledge brokerages.

4.2. Mailing list network

In this section, we discuss how to construct an affiliation network to show how the 136 posters are affiliated with the lists by virtue of their contributions. Social network analysis (SNA) serves as a vital tool in identifying knowledge brokers, and developing understanding of how they collaborate in the respective lists. Using SNA visualization techniques we reveal the affiliations or ties posters have with lists and see the patterns of their contributions in the lists. The pattern of posters' activities would have been difficult to observe from the numerical data presented in Tables 1 and 2. In the affiliation network, both knowledge seekers and knowledge providers share a space or list and what links them is the email messages they exchange. As shown in Fig. 3, lists form nodes (represented by squares) or points on a bipartite graph and two or more nodes are linked by a line if a poster (represented by spheres) posts email messages to both lists.

Building the mailing list network proceeded in two stages. In the first stage, information on the 136 posters was used to generate *posters-by-lists* data matrix **P**. Excerpt of the data matrix is shown in Table 3.

The matrix describes the affiliation a poster has with a node or list. This is analogous to a person belonging to a group ('persons×groups') or persons attending an events ('persons×events'). In matrix **P**, each cell P_{ij} indicates that the ith poster posted a certain number of email messages in the jth list. The data was used to generate an UCINET

Table 3
Data Matrix with posters names anonymised

Posters	Kde	Mentors	User
deb006	1	5	41
deb010	1	9	11
deb033	8	7	8
deb041	1	14	2
deb046	1	1	116

DL file type[1]. UCINET is a social network analysis software developed by Analytic Technologies, Inc. [6]. The second stage involves the visualization of the network. NetDraw, a component of UCINET, was used to visualize how the nodes are linked by the postings. All the 136 posters produced a cumbersome visualization. As [27] noted, when a social network grows in size, it becomes difficult to understand the visualization because the diagram becomes more complicated, with an increasing number of edges. One approach to this problem is to use the 'Transform > Dichotomize' function available in UCINET and decide on a cut-off value. The rule for this procedure is:

$$\text{Rule}: y(i,j) = 1 \quad \text{if } x(i,j) \geq 10, \text{ and 0 otherwise.}$$

This means that only participants who posted 10 or more email messages are shown in the visualization (Fig. 4.).

Those who posted less than 10 email messages were excluded and are shown at the top left-hand side of the visualization as excludes. The visualization shows that posters could be located at more than one node in the network and are capable of sharing their knowledge with other participants in other nodes. The exchange of email messages was used to establish ties between the nodes [10,15]. The mailing lists network enabled the identification of three groups or structures within the lists.

(1) *Group one.* Consider, for example, posters *deb367* in the Kde list, *deb141* in the Mentors list and *deb422* in the User list. These posters posted more

[1] The full data matrix (data-136 posters.txt), dl file (136 posters.##d)and the dichotomized matrix (136 posters-dichotomized matrix.##H) are available on request.

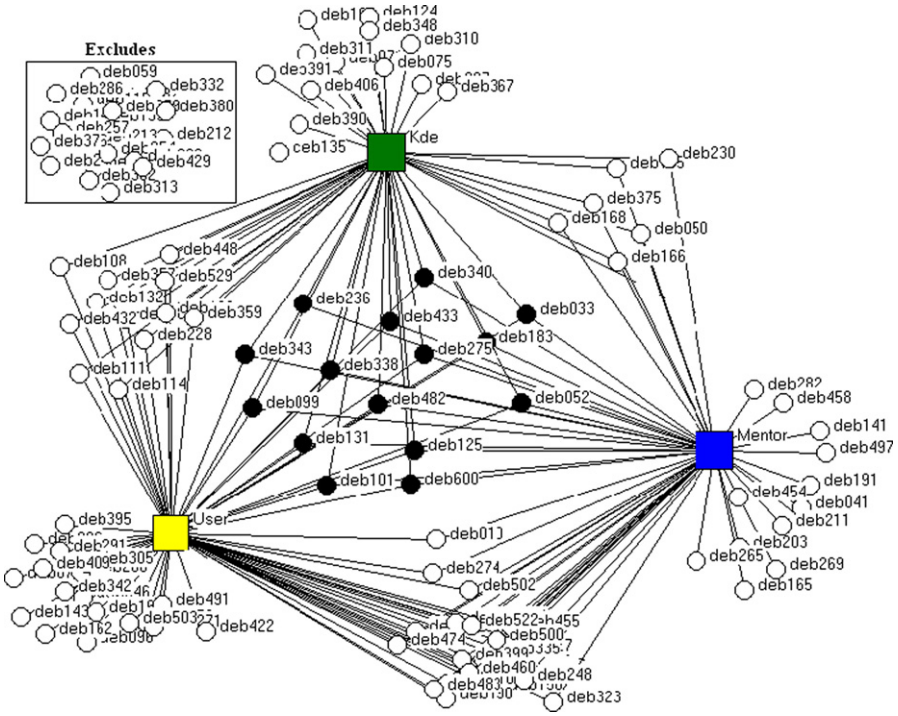

Fig. 4. Mailing list network.

information and/or provided more assistance within the lists they participated. Both knowledge seeking and knowledge providing are localized in these groups.

(2) *Group two.* Posters who participated across lists, externalizing their knowledge and expertise and helping answer questions knowledge seekers posted. For example;

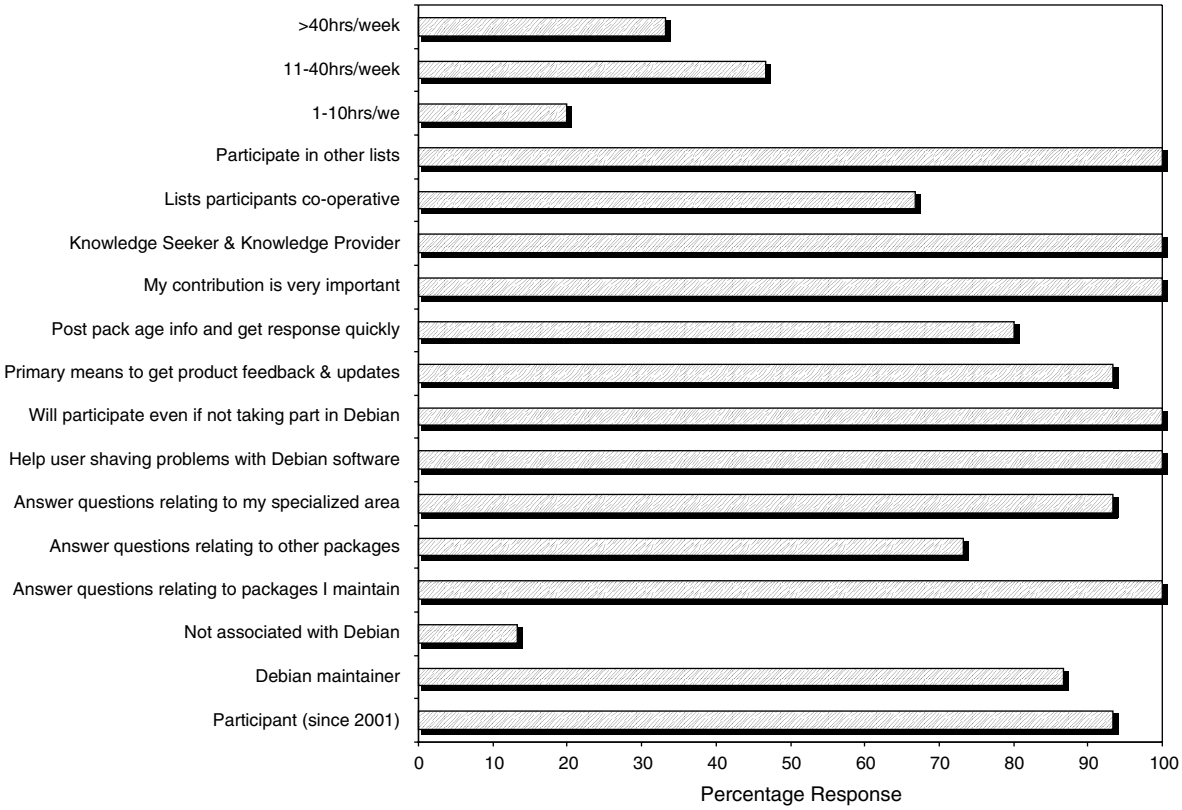

Fig. 5. Knowledge brokers' views on mailing lists.

(a) User∪Kde = {*deb111,deb114,...*}
(b) Kde∪Mentors = {*deb230,deb050,...*}
(c) Mentors∪User = {*deb010,deb274,...*}

(3) *Group three.* At the center of the visualization is a group of 15 posters (e.g. *deb600)* we call knowledge brokers. They link all the three nodes in the network. By linking and collaborating with list participants, they serve an important function as community facilitators or hubs [3] on the network. Barbasi [4] (p. 64) argued that hubs are special and paying attention to them is well deserved. They dominate the structure of all networks in which they are present, creating trends and fashions, making important deals, etc.

To complement our findings, the 15 knowledge brokers were invited (using their emails obtained from the data) to complete a questionnaire. The survey was conducted between November 14th and December 16th, 2005. Our aim was to find out what motivates the knowledge brokers to participate in the lists, how they view their roles as knowledge seekers and knowledge providers, and their perceptions of other participants, etc. Fig. 5 shows the results of the survey.

The knowledge brokers are long time (since 2001) list participants (93.3%, $N = 14$). Most of them are maintainers (86.7%, $N = 13$) and, therefore, are better positioned to help users having problems with Debian software (100%, $N = 15$). They see mailing lists as the primary means to get feedback and updates on their software packages (93.3%, $N = 14$) because they can post patches and information and get responses almost quickly (80.0%, $N = 12$). As we have identified them as hubs in our network, knowledge brokers recognize that their contributions to the project as a whole and the lists in particular are very important (100%, $N = 15$). They also see their roles as both knowledge seekers and knowledge providers (100%, $N = 15$). Their positive response may be due to the fact that they usually answer questions relating to packages they maintain (100%, $N = 15$) or anything relating to their specialized area (93.3%, $N = 14$). Yet, they answer questions relating to other packages (73.3%, $N = 11$) maintained by others. On average, knowledge brokers spend reasonable time (11–40 h/week) in lists (46.7%, $N = 7$) helping answer questions and posting information. They find other participants co-operative (66.6%, $N = 10$). Even though they might not be associated with the Debian project (13.3%, $N = 2$), the majority feels that they will continue to take part in these lists (100%, $N = 15$). They participate in other lists outside Debian (100%, $N = 15$) as well.

5. Conclusion

In this paper, we have employed a methodology to extract data from three mailing lists from the Debian project. We showed how knowledge seekers and knowledge providers interact in the lists. We counted the total numbers of posts each individual contributed and identified posters to all the three lists. Social network visualization technique was used to identify knowledge brokers within the lists. The visualization makes the structural analysis of three communities possible by enabling us to discern who the major contributors are. We highlighted regions in the network where 15 knowledge brokers are located. This region, although small, is surrounded by thin areas where knowledge seekers and knowledge providers exchange messages. A supplementary survey shows, among other measures, that knowledge brokers have a dual role as knowledge brokers and knowledge providers, perform an important role in the project, and will continue on these roles even when they cease to participate in the project. The ecology of lists' communities and the manner in which participants exchange emails has great significance when it comes to the overall organization and coordination of resources in OSS projects. We posit that many successful OSS project may have ample of these knowledge brokers. But, giving the nature of discussions in lists and the manner in which email messages are threaded (following next-to-next pattern), identifying knowledge brokers within the array of lists participants remain difficult. In this regard, this paper serves as an additional mechanism OSS project coordinators can use to help identify active and valuable expert human resources, vis-*a*-vis, knowledge brokers.

6. Limitation and further research

The major limitation of the work is in generalizing the findings we reported here. Our study utilized a tiny fraction of the Debian lists. The 136 posters used to identify the knowledge brokers contributed about 5.9% of the total number of email messages posted to the three lists. We plan to replicate the study using 12 more lists. This study is part of our ongoing work to developing a social network visualization tool. The tool will be used as a plug-in to enable mailing list users to visualize community participation (like the mailing list network presented here). Project hosts can see who is doing what, how much of it and in real-time see when a volunteer becomes inactive. We also conjectured a point worth further investigation—a situation where knowledge brokers suddenly stop contributing or resign from the project. In the mailing list network, this will tantamount to removing knowledge brokers one at a time until none is left. Then what remains is a fragmented network or sub-networks of the individual lists. Further research is needed to investigate how the fragmentation of the affiliation network affects knowledge seeking and knowledge providing in OSS projects, and OSS projects' success in general.

References

[1] T. Adelstein, Desktop Linux: New Linux users changing the face of community, DesktopLinux.com, Available from: http://www.desktoplinux.com/articles/AT3791991696.html.

[2] J.M. Barahona, L. Lopez, G. Robles, Community structure of modules in the Apache project, Proceedings of the Fourth Workshop on Open Source Software Engineering, Edinburgh, Scotland, 2004.
[3] A. Barbasi, E. Bonabeau, Scale-free networks, Scientific America 288 (5) (2003) 50–59.
[4] A. Barbasi, LINKED, The New Science of Networks, Cambridge, MA, 2002.
[5] A. Bonaccorsi, C. Rossi, Why Open Source Can Succeed, Available from: http://opensource.mit.edu/papers/rp-bonaccorsirossi.pdf.
[6] S.P. Borgatti, M.G. Everett, L.C. Freeman, Ucinet for Windows: Software for Social Network Analysis, Analytic Technologies, Harvard, MA, 2002.
[7] J. Feller, B. Fitzgerald, K. Lakhani, Collaboration, conflict and control, fourth workshop on open source software engineering, SIGSOFT Software Engineering Notes 30 (3) (2005) 1–2.
[8] B. Fitzerald, A critical look at open source, IT Systems Perspectives 10 (2004) 92–94.
[10] R.A. Ghosh, Clustering and dependencies in free/open source software development: methodology and tools, Firstmonday 8 (4) (2003).
[11] E. Giacoletto, K. Aberer, Automatic Expansion of Manual Email Classifications Based on Text Analysis, Springer, Berlin, 2003, pp. 785–802.
[12] P. Gloor, R. Laubacher, S. Dynes, Y. Zhao, Visualization of communication patterns in collaborative innovation networks analysis of some W3C working groups, ACM CKIM International Conference on Information and Knowledge Management, New Orleans, 2003, pp. 56–60.
[13] C. Gutwin, R. Penner, K. Schneider, Group awareness in distributed software development, Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work, Chicago, 2004, pp. 72–81.
[14] A. Hemetsberger, C. Reinhardt, Sharing and creating knowledge in open-source communities, The Case of KDE, Fifth European Conference on Organizational Knowledge, Learning, and Capabilities, Innsbruck, Austria, 2004, Available from: http://opensource.mit.edu/papers/hemreinh.pdf.
[15] R. Hanneman, Introduction to social network methods, in: S.P. Borgatti, M.G. Everett, L.C. Freeman (Eds.), Ucinet for Windows, Software for Social Network Analysis, Analytic Technologies, Harvard, MA, 2002.
[16] G. Hertel, S. Niedner, S. Herrmann, Motivation of Software Developers in Open Source Projects: An Internet-based Survey of Contributors to the Linux Kernel, Available from: http://opensource.mit.edu/papers/rp-hertelniednerherrmann.pdf.
[17] Q. Jones, G. Ravid, S. Rafaeil, Information overload and the message dynamics of online interaction spaces: a theoretical model and empirical exploration, Information System Research 15 (2) (2004) 194–210.
[18] K. Keith, The Ten Worst Engineering Pitfalls, OSNews.com, Available from: http://www.wetworx.com/posts/00000033.html.
[19] S. Koch, G. Schneider, Effort, cooperation and coordination in an open source software project: Gnome, Information Systems Journal 12 (1) (2002) 27–42.
[20] G. Krogh, S. Spaeth, K. Lakhani, Community, joining, and specialisation in open source software innovation: a case study, Research Policy 32 (2003) 1217–1241.
[21] K. Lakhani, E. Hippel, How open source software works: 'free' user-to-user assistance, Research Policy 32 (2003) 923–943.
[22] F.G. Lanzara, M. Morner, The Knowledge Ecology of Open-Source Software Projects, 19th EGOS Colloquium, Copenhagen, 2003.
[23] J. Lerner, J. Tirole, The open source movement: key research questions, European Economic Review 45 (5) (2001) 819–826.
[24] G. Madey, F. Freeh, R. Tynan, Modelling the free/open source software community: a quantative investigation, in: S. Koch (Ed.), Free/Open Source Software Development, Idea Group Inc., Hershey, PA, 2004, pp. 203–220.
[25] M. Michlmayr, Managing volunteer activity in free software projects, Proceedings of the 2004 USENIX Annual Technical Conference, Freenix Track, 2004, pp. 93–102.
[26] A. Mockus, R. Fielding, J. Herbsleb, Two case studies of open source software development: Apache and Mozilla, ACM Transactions on Software Engineering and Methodology 11 (3) (2003) 1–38.
[27] P. Mutton, Inferring and Visualising Social Networks on Internet Relay Chat, IEEE Computer Society, Washington, DC, 2004, p. 35–43.
[28] M.D. Nichols, B.M. Twidale, Usability and open source software, Firstmonday 8 (1) (2003).
[29] S.E. Raymond, The Cathedral and the Bazaar, O'Reilly, Sebastopol, CA, 1999.
[30] S.K. Sowe, A. Karoulis, I. Stamelos, A constructivist view of knowledge management in open source virtual communities, in: A.D. Figueiredo, A.P. Afonso (Eds.), Managing Learning in Virtual Settings: The Role of Context, Idea Group Inc., Hershey, PA, 2005, pp. 290–308.
[31] J. Xu, Y. Gao, S. Christley, S. Madey, A topological analysis of the open source software development community, IEEE Proceedings of the 38th Hawaii International Conference on System Sciences, Track 7, 2005, p. 195.
[32] J. Zawinski, Message Threading, Available from: http://www.jwz.org/doc/threading.html.