8-2007

Measuring knowledge sharing in open source software development teams

Y. LONG

Keng SIAU
Singapore Management University, klsiau@smu.edu.sg

K. HOWELL

Citation

LONG, Y.; SIAU, Keng; and HOWELL, K.. Measuring knowledge sharing in open source software development teams. (2007). *Proceedings of the 13th Americas Conference on Information Systems (AMCIS 2007), Keystone, Colorado, USA, August 9-12*. 1-12.
Available at: https://ink.library.smu.edu.sg/sis_research/9419

Measuring Knowledge Sharing in Open Source Software Development Teams

Yuan Long
Hasan School of Business
Colorado State University-Pueblo
yoanna.long@colostate-pueblo.edu

Keng Siau
College of Business Administration
University of Nebraska Lincoln
ksiau@unl.edu

Kris Howell
Hasan School of Business
Colorado State University-Pueblo
kris.howell@colostate-pueblo.edu

Abstract

The study provides an approach to measure the extent of knowledge sharing in Open Source Software (OSS) development in terms of two aspects: the quality of knowledge sharing which is indicated by the helpfulness of the messages, and the quantity of knowledge sharing which is indicated by the volume of the messages. The study developed a computer-aided content analysis program to assess the helpfulness of the messages based on a set of keywords and the length of the messages. The approach was applied to measure the extent of knowledge sharing of 150 OSS projects. The results further confirmed that the two measures (i.e., helpfulness of messages and volume of messages) assess different aspects of knowledge sharing. Another contribution of this research is the computer-aided content analysis program, which has been shown to be effective and reliable and can be applied in future research.

Keywords: Knowledge sharing, content analysis, Open Source Software (OSS) development

Introduction

Open Source Software (OSS) development has received increasing attention from both researchers and practitioners in last a few years. OSS refers to the computer software whose source code is open to the public (OpenSourceInitiative 2006). Anyone who is interested in the software is able to access, modify, customize, and redistribute the software under an open source license (OpenSourceInitiative 2006).

Most of the participants of OSS projects are volunteers and distributed all over the world (Long and Siau 2007). They communicate and collaborate with each other through the Internet. Therefore, effective and efficient knowledge sharing is critical to the success of OSS projects. In order to study the antecedents (e.g., what leads to effective knowledge sharing) and the consequences (e.g., the effects) of knowledge sharing, it is essential to find a feasible approach to measure the knowledge sharing of OSS teams. Using computer-aided content analysis, this paper provides a creative approach to measure the knowledge sharing of OSS teams based on two aspects: quality and quantity of knowledge sharing.

The paper is organized as follows: the next section presents a literature review on knowledge sharing, virtual communities, and previous measurement of knowledge sharing. The following section introduces the research method, sample selection, and data collection. Next, the paper discusses the steps to measure knowledge sharing in OSS teams. Research results are then provided. The final section summarizes the research and discusses future research directions.

Literature review

Knowledge sharing in virtual communities

Knowledge has been recognized as the primary resource of organizations (Alavi 2001, Argote 2003). Knowledge sharing in different organizational environment contributes to the success of organizations. With the advancement of telecommunication technology, virtual communities have become popular all over the world (Koh and Kim 2004). Virtual communities refer to physically distributed individuals or business partners who interact around common interests via electronic based communications which are guided by some protocols or norms (Porter 2004). Figure 1 shows a topology of virtual communities developed by Porter (2004).

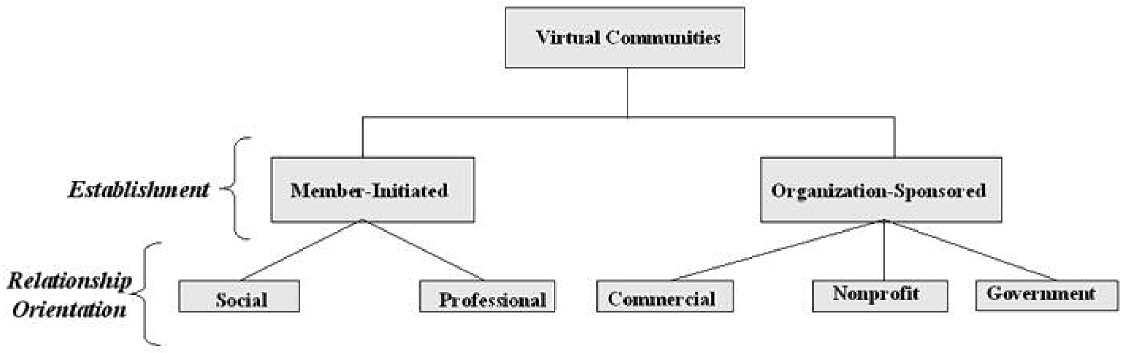


Figure 1. A typology of virtual communities (Porter 2004)

Open Source Software (OSS) development is a typical case of virtual communities. The participants of OSS projects are self-selected based on their common interests and are self-organized in terms of their contributions to the projects (Raymond 2000). OSS teams are member-initiated groups and the participants of OSS projects are professionals in specific computer fields (as shown in Figure 1). The participants communicate and collaborate with each other through the Internet. They share information and knowledge via emails or discussion boards. Therefore, an OSS community can be understood as a knowledge community via computer-mediated communication. Needless to say, efficient and effective knowledge sharing is critical to the success of OSS projects.

Prior literature has recognized the importance of knowledge sharing in OSS development. However, knowledge sharing in OSS projects remains very much an unstudied area. One obstacle to this area of research is an efficient and effective way of measuring knowledge sharing. Once an effective and efficient measure of knowledge sharing is available, the researchers will be able to empirically study the antecedents and the consequences of knowledge sharing. The next section reviews the assessment of knowledge sharing in the earlier literature.

Measuring knowledge sharing

Knowledge sharing refers to sharing information and knowledge among individuals, groups, and organizations. In this research we are interested in measuring the extent (the level) of knowledge sharing in OSS groups. It is not the intention of this research to classify knowledge into different categories (e.g., such as software versus hardware, and tacit versus explicit knowledge). Also, the research does not evaluate the attitudes towards knowledge sharing or the intention to share knowledge (Bock et al. 2005, Constandt et al. 2001). The focus of this research is to assess the extent of the knowledge sharing itself rather than the satisfaction of knowledge sharing experience (Bock et al. 2005) or the likelihood to share knowledge (Constandt et al. 2001).

Based on the literature review, we identified two primary approaches to measure knowledge sharing in organizations: perception-based method and activity-based method (Table 1).

Table 1. Perception-based measure and activity-based measure

	Perception-based measure	Activity-based measure
Explanation	Perception of knowledge sharing	Actual activities of knowledge sharing
Measurement	Survey	Frequency measure, content analysis
Related studies	Cumming (2004) Cross & Sproll (2004) Sarker et.al. (2002)	Bauer & Scharl (2000) Huang & DeSanctis (2005) Koh & Kim (2004) Wasko & Faraj (2005)

Questionnaire survey has been employed as a popular method to assess knowledge sharing, especially in traditional organizations. Generally, the researchers designed a list of questions and asked participants about their perception of the level of knowledge sharing (Cumming 2004, Cross and Sproll 2004, Sarker et al. 2002). In traditional organizations, people have various ways to share knowledge. They can share information through either face-to-face meeting or online document exchange. It is hard to record all the knowledge sharing related activities. Therefore, asking the individuals on their perception of the knowledge sharing can be an effective way to assess knowledge sharing.

In virtual communities (or virtual organizations), people communicate and collaborate with each other through electronic-based media. The knowledge sharing is through the Internet and in some cases the knowledge sharing process has been automatically recorded. This enables researchers to directly assess the extent of knowledge sharing based on the records of knowledge sharing activities. Table 2 summaries a few measures from relevant studies.

Table 2. Measurement of knowledge sharing

Articles	Target variables	Measures	Explanations
Borgatti & Cross (2003)	Information seeking	- Survey - Social metric method - Individual level	- Indicate how often the person has turned to you (or you have turned to the person) for information or knowledge (0-5 scale)
Huang and DeSanctis (2005)	Information exchange	- Content analysis (coding) - Group level (different forums)	Code the messages into the following four categories: - Information seeking and information providing - Explicit knowledge and tacit knowledge
Hansen (2002)	Amount acquired knowledge sharing	- Survey - Multiunit firms	- Indicate the percentage of the ware[1] acquired from other divisions
Koh & Kim (2004)	Knowledge sharing activity	- Frequency measure (archive data) - Group level (different virtual communities)	-Knowledge posting activity (mean number of posts/views per month for each individual community) -Knowledge viewing activity
Tsai (2002)	Intra-organizational knowledge sharing	- Survey - Intra-organizational units	- Indicate the units (a list of all the units in the company) from which the respondents received knowledge
Wasko & Faraj (2005)	Knowledge contribution	- Content analysis (coding) - Counting measure - Individual	- Helpfulness of contribution - Volume of contribution

The purpose of this study is to measure the extent of knowledge sharing in OSS development. The analysis is at the group level. With data (emails, messages, and logs) provided by the OSS project website (i.e., Sourceforge.net), we are able to directly assess the knowledge sharing through content analysis and frequency measure.

Research methods

Sample selection

Project samples were selected from SourceForge.net. SourceForge is the world's largest website hosting OSS projects. At the time of this study (October 2005), SourceForge hosted over one hundred thousand projects and involved over one million registered users[2]. Because of the rich and open information provided by SourceForge, it has become a popular data source for researchers. Although SourceForge may not capture all the OSS projects (a few OSS projects, especially those large projects such as Linux, have their own websites), it is probably the most popular data resource for OSS researchers to collect large sample size across different OSS groups (Madey 2002, Crowston and Howison 2005, Long 2005, Long and Siau 2007).

Stratified sampling was used in the study. Software categories provided by SourceForge were employed as the strata. Stratified sampling was used for several reasons. First, the sampling covers every software category, which ensures a better coverage of the population than simple random sampling. Second, stratified sampling generally has more statistical precision than random sampling. Projects within one category may share similarities which distinguish them from those in other categories. Therefore, the sampling ensures adequate variance across different types of projects. Third, stratification is convenient for the study. SourceForge divides the projects into different categories in terms of their major functions. In addition, the project in each category has a unique ID, which enables the random sampling within each category to be easily conducted.

[1] Ware here refers to "software and hardware that the focal team obtained from other divisions to solve emerging problems" (Hansen 2002)
[2] The exact number of registered projects was 104,078, and the exact amount of registered users was 1,155,179, as reported on October 12, 2005.

In each software category, the size of the sample is basically taken in proportion of the stratum. For example, suppose the total sample size is 150 projects. If the projects in the category on *Communication and Internet* occupy 20% of the total number, a number of 30 (150*20%=30) projects will be drawn from this category. After determining the sample size of each category, the study randomly selected projects within different categories (strata).

If the selected project met the following two criteria, it was then included into the sampling pool.

First, the project has to have an open bug tracking system. The bug tracking system is a forum enabling users and developers to report and discuss bugs (errors or inconsistencies of the source code). It is the primary data source of our study. Most projects open their bug tracking system to the public to encourage participation. However the bug tracking system is not always available. A few projects either restrict the forum only to the developers or use other tools such as mailing lists to track bugs instead of the bug tracking systems. Both of these situations make the data inaccessible.

Second, the project has to involve at least fifty bug reports in the bug tracking systems. The bug tracking system is one of the major places for OSS developers to communicate and collaborate with each other (Raymond 2000). The total number of bug reports indicates the degree of activity of the projects. Fifty bug-reports were chosen as a minimum to ensure sufficient interactions and information sharing of a selected project.

Not all projects on SourceForge are suitable for this study. A few projects do not have an open bug tracking system; while others are "dead" projects, which stay inactive with few activities and updates.

The above two criteria ensure that the selected projects are relatively active with acceptable levels of knowledge sharing activities among participants.

Table 3 lists the descriptive information for the project samples, which were collected from November to December 2005. At the time of the study, SourceForge listed around twenty software categories. In an effort to retain parsimony, these topics were combined into seven primary categories with an extra one named "others". The category "others" includes categories with a small number of projects, such as religion and philosophy, sociologies, and terminals.

Table 3. Descriptive information for sample

Software Category	Number of Projects Sampled	Number of Developers			Project Tenure[1] (Month)		
		Min	Max	Mean[2]	Min	Max	Mean[2]
Database	8	5	91	7	20	69	55
Games/Entertainment	20	3	68	20	17	72	48
Communication/Internet	30	4	325	35	7	74	53
Office business	17	3	46	13	7	69	37
Scientific/Engineering	31	2	45	12	6	72	42
Software development	22	2	55	17	11	72	45
System	11	2	36	18	16	74	43
Others	11	3	67	21	26	74	57
Summary	**150**	**2**	**325**	**21**	**6**	**74**	**47**

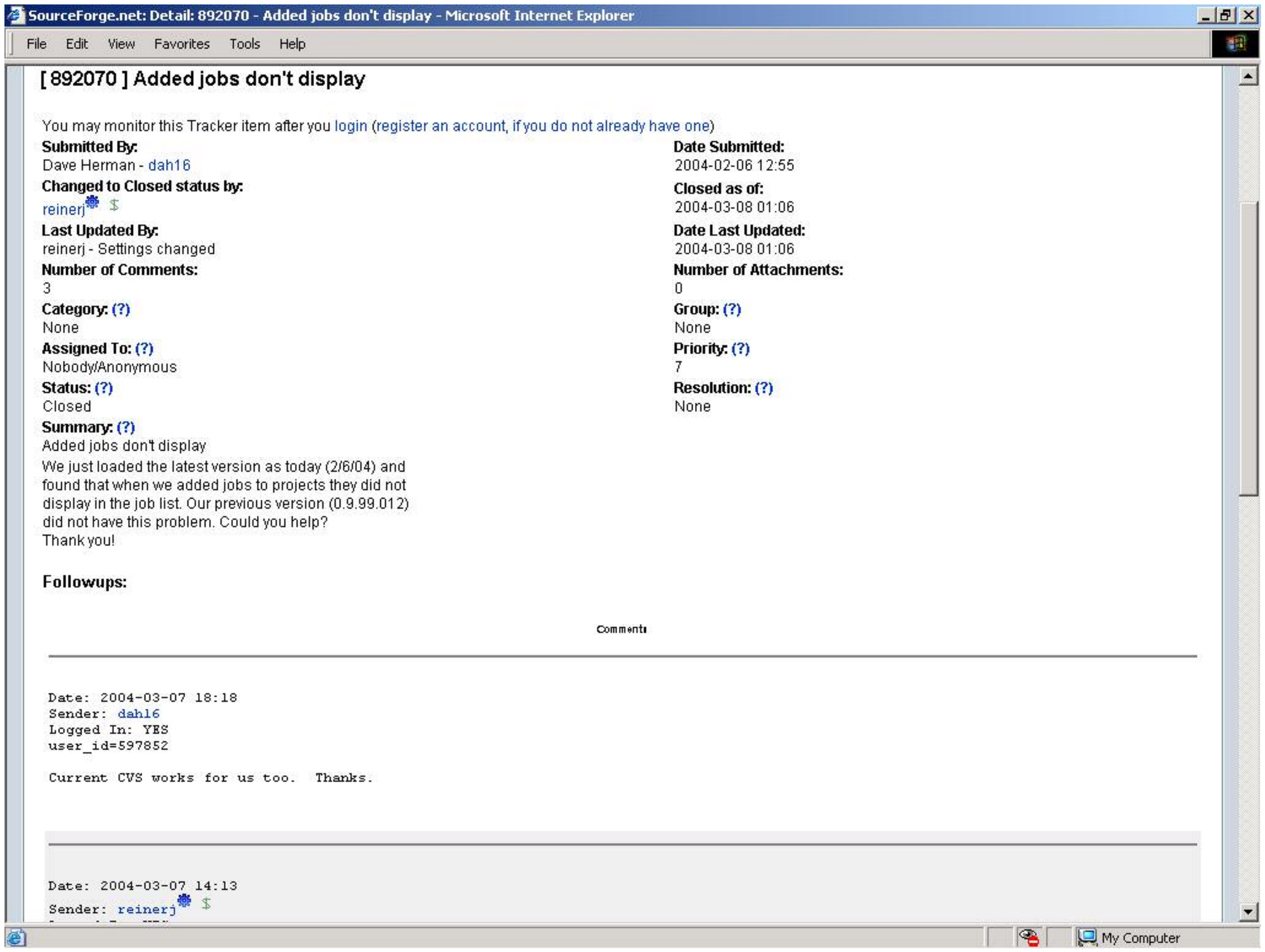
[1] Counted from the project registered date to the date of the study
[2] Rounded to nearest number

The above table suggests a large variety of OSS projects. First, the samples cover all the software categories listed on SourceForge, which ensures better coverage of the population and adequate variance across different types of projects. Second, the number of developers ranges widely from 2 to 325, which indicates a variety of group size. Third, the project tenure varies from 6 to 74 month, which suggests the history of the projects spans from a minimum of half a year to a maximum of six years.

Data collection

The data were gathered from the bug tracking system of each selected project. The bug tracking system enables participants, including both developers and users, to report, discuss, and track bugs. In the bug tracking system, somebody reports a bug and others reply to the message with discussions and solutions (as shown in Figure 2). In addition to the discussion board, the system also provides the following information to facilitate bug tracking (as shown in Figure 3).

- Bug summary, including the content of the bug, the open date, the person who submits the bug, and the person to whom the bug is assigned
- Status of the bug, including four statuses, open, closed, deleted, and pending
- Priority of the bug, from 1 to 9, indicating the least to the most important.


SourceForge.net: Detail: 892070 - Added jobs don't display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
My Computer

[892070] Added jobs don't display

You may monitor this Tracker item after you login (register an account, if you do not already have one)

Submitted By: Dave Herman - dah16	**Date Submitted:** 2004-02-06 12:55
Changed to Closed status by: reinerj	**Closed as of:** 2004-03-08 01:06
Last Updated By: reinerj - Settings changed	**Date Last Updated:** 2004-03-08 01:06
Number of Comments: 3	**Number of Attachments:** 0
Category: (?) None	**Group:** (?) None
Assigned To: (?) Nobody/Anonymous	**Priority:** (?) 7
Status: (?) Closed	**Resolution:** (?) None

Summary: (?)
Added jobs don't display
We just loaded the latest version as today (2/6/04) and found that when we added jobs to projects they did not display in the job list. Our previous version (0.9.99.012) did not have this problem. Could you help?
Thank you!

Followups:

Comments

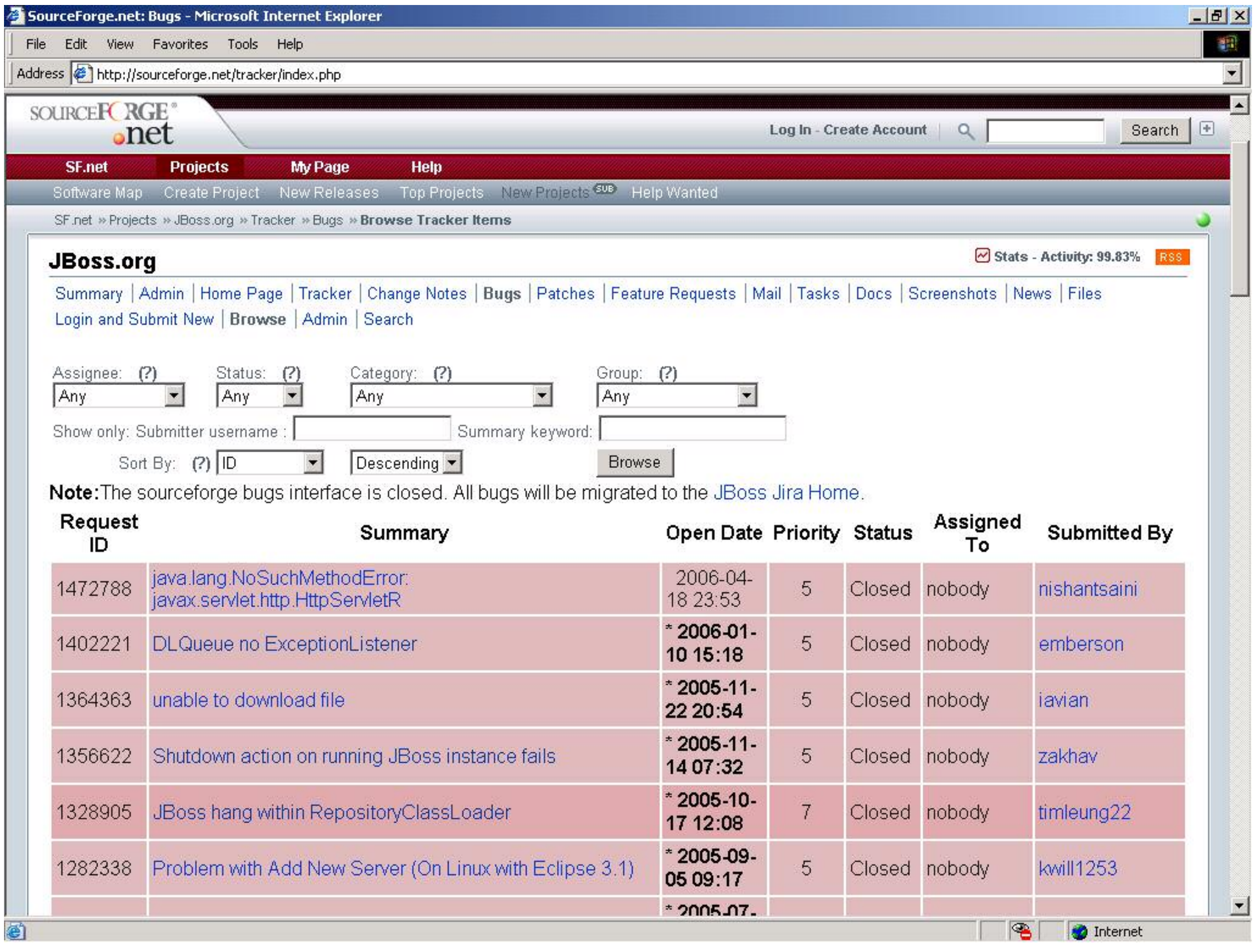
```
Date: 2004-03-07 18:18
Sender: dah16
Logged In: YES
user_id=597852

Current CVS works for us too.  Thanks.
```

```
Date: 2004-03-07 14:13
Sender: reinerj
```

Figure 2. An example of a bug report

We chose bug tracking system as the primary data source for the following reasons. First, OSS development is characterized as peer review of open codes. Raymond (2000) proposed the "Linux' law" in his well known essay "Cathedral and the Bazaar", "Given enough eyeballs, all bugs are shallow." Therefore, the bug tracking system is a key feature of OSS development as well as an important venue for participants (both developers and users) to collaborate with each other. Second, compared to other development activities such as feature request, the bug-fixing process is the most active procedure that reveals close collaboration and rich interaction. Third, the majority of the posts on bug tracking systems center around bug reports and bug solving. The topics are much more focused than other forums such as mailing lists. In mailing lists, people can chat on whatever topics they desire, which may not be relevant to knowledge sharing in software development. Lastly, most OSS projects open their bug tracking systems to the public, which can be accessed by academic researchers. These bug tracking systems record the historical threads since the beginning of the projects. Some other communication tools such as maligning lists are not always open to the public but only to those registered participants. In addition, group meetings through MSN messenger[3] or Skype[4] are neither recorded nor open to the public.

In the study, the entire bug-report Webpages of each selected project were downloaded. A total of 76,992 bug reports were incorporated in the study, with an average of 513 bug reports per project. Each Webpage (a bug report) includes multiple messages, which were used as the data source for the content analysis.


SourceForge.net: Bugs - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://sourceforge.net/tracker/index.php
SOURCEFORGE.net
Log In - Create Account
Search
SF.net Projects My Page Help
Software Map Create Project New Releases Top Projects New Projects Help Wanted
SF.net » Projects » JBoss.org » Tracker » Bugs » Browse Tracker Items
JBoss.org
Stats - Activity: 99.83% RSS
Summary | Admin | Home Page | Tracker | Change Notes | Bugs | Patches | Feature Requests | Mail | Tasks | Docs | Screenshots | News | Files
Login and Submit New | Browse | Admin | Search
Browse
Internet

Assignee: (?) Any Status: (?) Any Category: (?) Any Group: (?) Any

Show only: Submitter username : Summary keyword:

Sort By: (?) ID Descending

Note: The sourceforge bugs interface is closed. All bugs will be migrated to the JBoss Jira Home.

Request ID	Summary	Open Date	Priority	Status	Assigned To	Submitted By
1472788	java.lang.NoSuchMethodError: javax.servlet.http.HttpServletR	2006-04-18 23:53	5	Closed	nobody	nishantsaini
1402221	DLQueue no ExceptionListener	* 2006-01-10 15:18	5	Closed	nobody	emberson
1364363	unable to download file	* 2005-11-22 20:54	5	Closed	nobody	iavian
1356622	Shutdown action on running JBoss instance fails	* 2005-11-14 07:32	5	Closed	nobody	zakhav
1328905	JBoss hang within RepositoryClassLoader	* 2005-10-17 12:08	7	Closed	nobody	timleung22
1282338	Problem with Add New Server (On Linux with Eclipse 3.1)	* 2005-09-05 09:17	5	Closed	nobody	kwill1253
		* 2005-07-				

Figure 3. An example of a bug tracking system

Research procedure in measuring knowledge sharing

Computer aided content analysis

Content analysis is a research method in social science to analyze communication content (Babbie 2006, Holsti 1969, Weber 1990). It has been actively used in different fields for more than 50 years to determine the presence of certain words or concepts in texts (Wagner et al. 2003, Busch et al. 2005). By analyzing and quantifying the presence and relationships of these words and concepts, researchers are able to make references about antecedents, characteristics, and consequences of the problems being studied (Holsti 1969).

In many instances, it is impossible to manually code all the information. Therefore, computer programs are needed to help us automatically code the information. Based on different functions, there are three major categories of computer-aided content analysis programs (Lowe 2002). The first set of programs conduct "dictionary-based content analysis" (Lowe 2002, P.1).

[3] MSN messenger is an Internet instant messenger service (Wikipedia)
[4] Skype is a peer-to-peer telephony network (Wikipedia)

These programs provide basic analysis functions such as word counting, sorting, and simple statistical tests. The second set of programs provide "development environment" (Lowe 2002, P.1). They facilitate the construction of analysis tools such as dictionaries and statistical tool. And the third set of programs include "annotation aids" (Lowe 2002, P.1) that work as an aid to researchers by providing functions such as marginal notes and cross-references.

The purpose of our study is to measure the extent of knowledge sharing in OSS development teams. We need to analyze and quantify the presence of words from thousands of messages. We did a research on the different types of content analysis programs but could not find an existing program that suits our specific requirements. Therefore we decided to develop our own program to analyze the messages.

Measurement

The extent of knowledge sharing was measured in terms of two aspects: the quality and the quantity of knowledge sharing.

Quality of knowledge sharing

The first measure assesses the quality of messages, which is indicated by the degree of helpfulness of messages. Content analysis was employed to determine the helpfulness of messages in terms of three levels: Very helpful, somewhat helpful, and not at all helpful. The coding policy is based on the work of Wasko and Faraj (2005). These three levels of helpfulness are defined as below.

- **Very helpful** (coded as 2). The response not only answers the question directly, but also provides explanations or references to relevant knowledge sources.
- **Somewhat helpful** (coded as 1). The response answers the question however provides little explanations. Or, the response does not answer the question directly but provides information relevant to the problem.
- **Not helpful** (coded as 0). The response is not related to the query. It can be a question itself, a social greeting (such as *Thank you* and *Hello*), or an announcement. In any of these cases, the response is not helpful in sharing knowledge.

It is not feasible to manually code a large amount of messages such as the messages posted in the bug tracking systems in OSS projects. Therefore, we developed a content analysis program to automatically code the messages.

The coding process is shown in Figure 4.

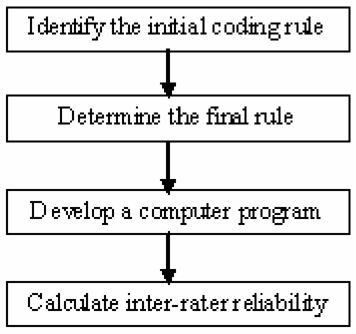

Identify the initial coding rule
Determine the final rule
Develop a computer program
Calculate inter-rater reliability

Figure 4. Coding process

<u>Step 1 Identifying the initial coding rule</u>

In order to develop a content analysis program, the first step is to identify the coding rule. The coding rule is guided by the above coding policy (i.e., three levels of helpfulness of messages). However this coding policy is merely a general guideline which cannot be executed by a computer program. Therefore, the general policy needs to be translated into specific coding rules that can be understood by a computer program.

To develop the coding rule, one hundred bug tracking pages were randomly selected (over 200 messages all together) from the sampling pool. Two human coders, who are experts in software development, were then asked to code each message independently based on the coding policy. The two coders rated each message carefully and recorded the reason for each decision. After comparing the coding results and discussing the discrepancies, they agreed on an initial coding rule. The rule was developed in terms of two aspects: a set of key words and the length of the message.

Choosing both the key words and the length of the messages to assess the helpfulness of the messages is based on two reasons. First, the discussion board (i.e., bug tracking systems) is a specific space for professionals such as programmers and developers to discuss bugs. The topics are much more focused than some other social forums. The basic theme of the discussion is concentrated on finding a bug, revising the code, and reporting the result. A few words related to the bug tracking such as add, create, and CVS are used frequently. Therefore, it is feasible to identify a set of key words that appear frequently and are closely related to the bug tracking.

However, the set of key words may not be able to completely cover all the messages that contribute to bug tracking. For example, in some situations the programmers post a paragraph of revised coding without any explanations. Since the bug tracking system is a professional forum, the assumption is that people come to the forum with common interests and a sufficient knowledge background to understand the coding. In other words, the coding help in knowledge sharing but it may not include those identified key words. In such cases, the coding is usually relatively long. Therefore, in addition to a set of key words, we also considered the length of the message as one component to measure the helpfulness of the messages.

Therefore, the initial rule includes two aspects: a set of keywords and the length of the messages. The keywords include verbs such as change, check, and modify, which are critical to discussing and solving a problem. They were evaluated via sentence analysis and frequency analysis. The appropriate message length was decided by observation and continual revision. For example, if a message contains at least two keywords and the length of the message is longer than fifty words, it is likely to include not only answers but also explanations. As a result, the message is coded as 2 (very helpful). If a message contains no key words and the length of the message is less than 10 words; most likely it is a social greeting or simply a reply irrelevant to the question.

Step 2 Determining the final rule

In order to ensure the accuracy of the coding rule, the two coders applied the initial coding rule to another one hundred bug tracking pages which were randomly selected from the sampling pool.

The two coders continuously revised the initial rule to resolve the disagreement between them. The selection of the keywords is based on both the frequency and the extent of relevance to bug tracking (The final list of the set of key words is available from the authors).

An inter-rater reliability between the two human coders was calculated after the coding. The reliability was accessed at 0.87 using Krippendorff's alpha statistic (Krippendorff 1980) and at 0.75 using Kappa's alpha statistic, which indicate a relatively high reliability of the coding. The final coding rule is based on the following matrix shown in Table 4.

Table 4. Coding rule

K.W.[2] \ L.M.[1]	0-20	20-50	>50
0	0[3]	0	1
1	1	1	2
>=2	1	2	2

[1] L.M.: Length of the Message
[2] K.W.: The number of Keywords appears in the message
[3] Coding. For example, if the length of a message is less than 20 words and contains no key words, it will be coded as 0.

Step 3 Developing a content analysis program

After determining the coding rules, the next step is to integrate the rule into a computer program. A content analysis program was developed by one author using Perl language (The program is available from the authors). Perl (an abbreviation for Practical Extraction and Report Language) was employed for its high capability in text processing, such as coding, libraries, and extensions (Wikimedia, 2006). It is a free software and widely used in text related tasks.

Step 4 Calculating the inter-rater reliability

To ensure the validity and reliability of the content analysis program, the two coders and the computer program simultaneously coded 100 bug tracking web pages that were randomly selected from the sampling pool. Minor revisions were made to solve the discrepancy between the two coders as well as between the coders and the computer program.

Table 5 shows the inter-rater reliability between the two coders as well as between the coders and the computer program. The results indicate a relatively high reliability of the coding process.

Table 5. Inter-rater reliability

	Coder 1	Coder 2	Computer program
Coder 1			
Coder 2	0.87/0.72[1]		
Computer program	0.72/0.68	0.71/0.68	

[1] Krippendorff's alpha / Kappa's alpha

Quantity of knowledge sharing

The second measure of knowledge sharing assesses the volume of messages, which indicates the quantity of knowledge sharing. The value was calculated using the mean number of messages (the number of messages posted minus the number of messages coded as 0) posted by the whole group for each month. A computer program was also developed to automatically determine the number.

RESULTS

The content analysis program has been applied to assess the knowledge sharing of 150 OSS projects. Table 6 shows the descriptive statistics (n=150) of both the quality and the quantity of knowledge sharing.

Table 6. Descriptive statistics of knowledge sharing

	Qualityof knowled ge sharing	Quantity of knowledge sharing
Mean	1.20	54.49
S.D.	0.16	68.90
Max	1.62	370.00
Min	0.65	7.00
Skewness	-.058	2.68
Kurtosis	0.219	6.861
K-S test	.200[1]	.000
Quality		-.037[2]
Quantity	-.037[2]	

[1] Significant at 0.05 level
[2] Correlation between quality of knowledge sharing and quantity of knowledge sharing

Based on the above results, we have the following observations. First, the correlation between the quantity of knowledge sharing and the quality of knowledge sharing is not significant (at 0.05 level). The result indicates that these two measures are independent of each other. Each measure assesses different aspect of the knowledge sharing. Second, the relationship between the quality and the quantity of knowledge sharing is negative (-0.037). The result is not surprising as a high quantity of knowledge sharing does not necessarily mean a high quality of knowledge sharing. Third, the distribution of the quality of the knowledge sharing (mean helpfulness of messages) is close to a normal distribution, while the distribution of the quantity of the knowledge sharing (average number of messages posted per month) is skewed (to the lower end). The skewed distribution of the volume of knowledge sharing might because of the skewed distribution of the group size. The average amount of messages posted by each group is related to the number of group members. In OSS development, a large amount of the projects have less than 5 developers.

Conclusion and future research directions

The research provides an approach to measure the extent of knowledge sharing of OSS teams in terms of two aspects: the quality of knowledge sharing and the quantity of knowledge sharing. A computer program was developed to automatically code the helpfulness of the messages based on a set of keywords and the length of the messages. Inter-rater reliability was calculated to ensure the reliability of the coding method.

The research has significant implications for information systems literature. The study provides a creative and feasible approach to automatically assess knowledge sharing of OSS teams. It helps researchers to better measure knowledge related constructs in virtual communities and virtual organizations.

Future research will focus on two directions. First, research needs to improve the measurement of knowledge sharing in OSS teams. There are some other communication tools for OSS participants to collaborate with each other, such as mailing lists and discussion forums. Future research may include these sources to compare between different communication patterns and knowledge sharing processes. In addition, other measurement techniques such as survey may be employed to complement the direct coding of messages.

Second, research needs to study the antecedents and the consequences of knowledge sharing in OSS development. Our research is ongoing; further analyses will be conducted to test the factors leading to or resulting from knowledge sharing in OSS development.

References

Alavi, M. a. D. E. L. "Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues," MIS Quarterly (25:1), 2001, pp. 107-136.

Argote, L., McEvily, B. and Reagans, R. "Managing Knowledge in Organizations: An Integrative Framework and Review of Emerging Themes," Management Science (49:4), 2003, pp. 572-582.Babbie, E. "The Practice of Social Research, 10th edition", Wadsworth: Thomson Learning Inc, 2006.

Bauer, C. and Scharl, A. "Quantitative evaluation of web site content and structure," Internet Research: Electronic Networking Applications and Policy, (10:1), 2000, pp. 31-43.

Bock,G., Zmud, R.W., Kim, Y., and Lee, J. "Behavioral intention formation in knowledge sharing: examining the roles of extrinsic motivations, social-psychological forces, and organizational climate," MIS Quarterly (29:1), March 2005, pp. 87-111.

Borgatti, S. P., and Cross, R. "A Relational View of Information Seeking and Learning in Social Networks," Management Science (49:4), 2003, pp. 432-445.

Busch, C., De Maret, P. S., Flynn, T., Kellum, R., Le, S., Meyers, B., Palmquist, P., Saunders, M., and White, R. Content Analysis. Writing@CSU. Colorado State University Department of English. Retrieved [3/1/2007] from http://writing.colostate.edu/guides/research/content/, 2005.

Cross, R., and Sproull, L. "More Than an Answer: Information Relationships for Actionable Knowledge," Organization science (15:4), 2004, pp. 446-462.

Crowston, K., and Howison, J. "The social structure of Free and Open Source software development," First Monday (10:2), 2005.

Cummings, J. N. "Work Groups, Structural Diversity, and Knowledge Sharing in a Global Organization," Management Science (50:3), 2004, pp. 352-364.

Hansen, M.T. "Knowledge Networks: Explaining Effective Knowledge Sharing in Multiunit Companies" Organization Science (13:3), 2002, pp.232-248.

Huang, S., and DeSanctis, G. "Mobilizing Informational Social Capital in Cyber Space: Online Social Network Structural Properties and Knowledge Sharing," Twenty-Sixth International Conference on Information Systems (ICIS 2005), Las Vegas, NV, 2005.

Holsti, O. R. "Content Analysis for the Social Sciences and Humanities," Reading, Mass, 1969.

Koh, J. and Kim Y. "Knowledge sharing in virtual communities: an e-business perspective," Expert Systems with Applications (26:2), 2004, pp. 155-166.

Krippendorff, E. "Content Analysis: An Introduction to its Methodology," Beverly Hills, CA: SAGE, 1980.

Long, Y. "Social Structure for Open Source Software Projects," Omaha, NE, AMCIS 2005, 2005.

Long, Y. and Siau, K. "Social Network Structures in Open Source Software Development Teams," Journal of Database Management (18:2), 2007, pp. 25-40

Lowe, W. "Software for content analysis: A review&apos; Report for the Identity Project," Retrieved from http://people.iq.harvard.edu/~wlowe/Publications/content.pdf, 2002.

Madey, G., Freeh, V., and Tynan R. "The Open Source Software Development Phenomenon:
An Analysis Based on Social Network Theory," AMCIS2002, Dallas, TX, 2002.

OpenSourceInitiative, "The Open Source Definition (V.1.9)," Retrieved from http://www.opensource.org/docs/definition.php. July, 2006

Porter, C.E. "A Typology of Virtual Communities: A Multi-Disciplinary Foundation for Future Research," Journal of Computer-mediated Communication (10:1), 2004.

Raymond, E. S. "The Cathedral and the Bazaar", Retrieved from http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/, 2000.

Sarker, S., Nicholson, D., and Joshi, K. "Knowledge transfer in virtual information systems development teams: an empirical examination of key enablers," Proceedings of 36th Hawaii International Conference on System Sciences (HICSS'03), 2002.

Tsai, W. "Social structure of "coopetition" within a multiunit organization: coordination, competition, and intraorganizational knowledge sharing," Organization science (13:2), 2002, pp.179-190.

Wagner W. P., Chung, Q. B., and Najdawi, M. K. "The Impact of Problem Domains and Knowledge Acquisition Techniques: A Content Analysis of P/OM Expert System Case Studies," Expert Systems with Applications (24:1), 2003, pp. 79-86.

Wasko, M. M., and Faraj, S. "Why Should I Share? Examining Social Capital and Knowledge Contribution in Electronic Networks of Practice," MIS Quarterly (29:1), 2005, pp. 35-57.

Weber, R. P., "Basic Content Analysis," 2nd ed., Newbury Park, CA: Sage, 1990.

Wikimedia. "Open-Source Software," Retrieved from http://en.wikipedia.org/wiki/Open_source_software, July, 2006.