

Singapore Management University

## Institutional Knowledge at Singapore Management University

---

Research Collection School Of Computing and Information Systems

School of Computing and Information Systems

---

4-2007

### Social network structures in open source software development teams

Y. LONG

Keng SIAU

Singapore Management University, klsiau@smu.edu.sg

Follow this and additional works at: [https://ink.library.smu.edu.sg/sis\\_research](https://ink.library.smu.edu.sg/sis_research)



Part of the [Databases and Information Systems Commons](#), and the [Digital Communications and Networking Commons](#)

---

#### Citation

LONG, Y. and SIAU, Keng. Social network structures in open source software development teams. (2007). *Journal of Database Management*. 18, (2), 25-40.

Available at: [https://ink.library.smu.edu.sg/sis\\_research/9379](https://ink.library.smu.edu.sg/sis_research/9379)

This Journal Article is brought to you for free and open access by the School of Computing and Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Computing and Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email [cherylds@smu.edu.sg](mailto:cherylds@smu.edu.sg).



# Social Network Structures in Open Source Software Development Teams

*Yuan Long, Colorado State University-Pueblo, USA*

*Keng Siau, University of Nebraska-Lincoln, USA*

---

## ABSTRACT

*Drawing on social network theories and previous studies, this research examines the dynamics of social network structures in open source software (OSS) teams. Three projects were selected from SourceForge.net in terms of their similarities as well as their differences. Monthly data were extracted from the bug tracking systems in order to achieve a longitudinal view of the interaction pattern of each project. Social network analysis was used to generate the indices of social structure. The finding suggests that the interaction pattern of OSS projects evolves from a single hub at the beginning to a core/periphery model as the projects move forward.*

*Keywords: longitudinal study; open source software (OSS); social networks; social structure*

---

## INTRODUCTION

The information system development arena has seen many revolutions and evolutions. We have witnessed the movement from structured development to object-oriented (OO) development. Modeling methods, such as data flow diagram and entity relationship diagram, are facing new OO modeling languages, such as the unified modeling language (UML) (see Siau & Cao, 2001; Siau, Erickson, & Lee, 2005; Siau & Loo, 2006) and OO

methodologies, such as unified process (UP). The latest development includes agile modeling (see Erickson, Lyytinen, & Siau, 2005), extreme programming, and OSS development. While many of these changes are related to systems development paradigms, methodologies, methods, and techniques, the phenomenon of OSS development entails a different structure for software development teams.

Unlike conventional software projects, the participants of OSS projects are volun-

teers. They are self-selected based on their interests and capability to contribute to the projects (Raymond, 2000). In addition, the developers of OSS projects are distributed all around the world. They communicate and collaborate with each other through the Internet, using e-mails or discussion boards. Therefore, effective and efficient communication and collaboration are critical to OSS success. However, little empirical research has been conducted to study the underlying interaction pattern of OSS teams, especially the dynamics of the social network structures in OSS development teams. To fill this gap, this study examines the evolvement of social structure in OSS teams. The study contributes to the enhancement of the understanding of OSS development, and provides foundation for future studies to analyze the antecedents and consequences of social networks in the OSS context.

The remainder of the paper is structured as follows. First, prior studies on social network structures in OSS teams are reviewed. Second, theories related to social structure and social network theory are discussed. Third, the research methodology is presented, and the research results are reported. Next, discussions of the results, the limitations, and the implications are provided. The paper concludes with suggestions for future research.

## LITERATURE REVIEW

The phenomenon of OSS development has attracted considerable attention from both practitioners and researchers in diverse fields, such as computer science, social psychology, organization, and management. Because of the multifaceted nature of OSS, researchers have investigated OSS phenomenon from varied perspectives. For

example, focusing on technical perspective, researchers studied issues such as OSS development methodology (e.g., Jørgensen, 2001) and coding quality (e.g., Stamelos, Angelis, Oikonomu, & Bleris, 2002). Based on social psychology, researchers investigated individual motivation (e.g., Hann, Robert, & Slaughter, 2004), new developers (Von Krogh, Spaeth, & Lakhani 2003), the social network (e.g., Madey, Freeh, & Tynan, 2002), and the social structure (e.g., Crowston & Howison, 2005). In terms of organizational and managerial perspective, researchers examined knowledge innovation (e.g., Hemetsberger 2004; Lee & Cole 2003, Von Hippel & von Krogh, 2003) and the governance mechanism (e.g., Sagers 2004).

An OSS development team is essentially a virtual organization in which participants interact and collaborate with each other through the Internet. Compared to conventional organizations, the structure of virtual organizations is decentralized, flat, and nonhierarchical (Ahuja & Carley 1999). However, some researchers challenge the belief (e.g., Crowston & Howison 2005; Gacek & Arief, 2004; Mockus, Fielding, & Herbsleb, 2000; Mockus, Fielding, & Herbsleb, 2002; Moon & Sproull, 2000). They argue that the social structure of OSS projects is hierarchical rather than flat, like a tree (Gacek & Arief, 2004) or an onion (Crowston & Howison, 2005). The social structure of OSS teams directly influences the collaboration and the decision-making process and further affects the overall performance of the teams as well as individuals' perception of belonging and satisfaction. Therefore, one wonders what form of social structure might be present in the OSS development and what type of structure will emerge—centralized or decentralized, hierarchical or nonhierarchical, onion-like

or tree-like, or a combination of the above depending on certain specific situations?

A social network, as stated by Krebs and Holley (2004), is generally built in four phases, each with its own distinct topology (as shown in Figure 1).

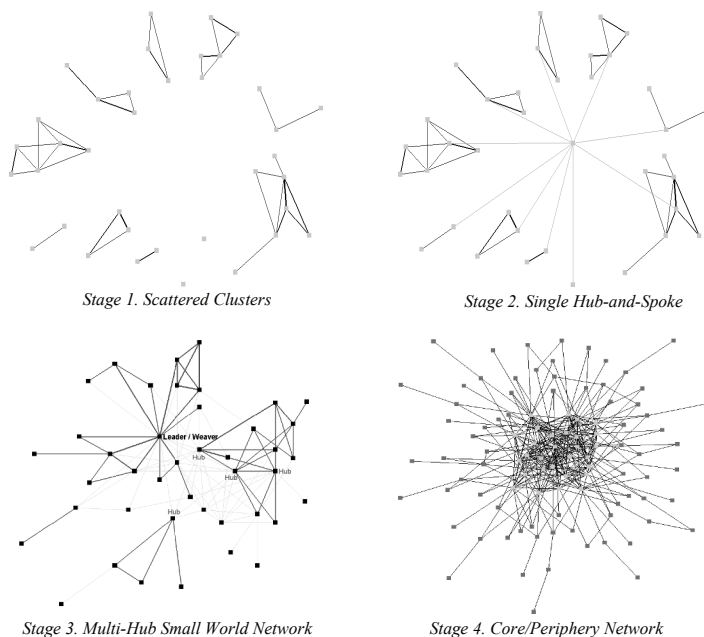
1. scattered clusters,
2. single hub-and-spoke,
3. multihub small-world network, and
4. core/periphery.

Most organizations start from isolated and distributed clusters (Krebs & Holley, 2004). Then an active leader emerges and takes responsibility for building a network that will connect the separate clusters. However, this single-hub topology is fragile. With more participants entering the group, the leader changes his/her role to a facilita-

tor and helps to build multiple hubs, which is stage three. The core/periphery model, the last stage, is the most stable structure. In the core/periphery model, the network core encompasses key group members who are strongly connected to each other, while the periphery contains members who are usually weakly connected to each other as well as to the core members. With the general network building phases in mind, one can argue that OSS projects may follow the same four stages (i.e., scattered clusters, single hub-and-spoke, multihub small-world network, and core/periphery model). But is that true for OSS projects? How does the social structure of OSS teams evolve over time?

Our research addresses the following two questions:

*Figure 1. Four phases of social structures (from Krebs and Holley 2004)*



1. What is the social structure of OSS teams?
2. How does the social structure evolve over time?

## **THEORETICAL FOUNDATION**

### **Social Structure and Social Interaction**

Social structure, as suggested by Schaefer and Lamm (1998), refers to the way in which society is organized into predictable relationships. Social structure can be considered in terms of three aspects—actors, their actions, and their interactions. The social actor is a relatively static concept addressing issues such as roles, positions, and statuses. Individual actors are embedded in the social environment and, therefore, their actions are largely influenced by the connections between each other. Social interaction is generally regarded as the way in which people respond to one another. These interaction patterns are to some extent independent of individuals. They exert a force that shapes both behavior (i.e., actions) and identity (i.e., actors) (Schaefer & Lamm, 1998).

Research on social interaction focuses on how individuals actually communicate with each other in group settings. These studies address issues such as the interaction patterns, the underlying rules guiding interaction, the reasons accounting for the way people interact, and the impacts of the interaction patterns on the individual behavior and the group performance. These issues begin by questioning what might be the interaction pattern in a specific social setting and that addresses our research question—understanding social interaction of OSS project teams.

### **Social Network Theory**

Social network theory focuses on studying actors as well as their relationships in specific social settings. Network theory is analogous to systems theory and complexity theory. It is an interdisciplinary theory stemming from multiple traditional fields, including psychology, which addresses individuals' perception of social structure; anthropology, which emphasizes social relationships; and mathematics, which provides algorithms (Scott, 2000).

Based on the view of social network, the world is composed of actors (also called nodes) and ties between them. The ties can represent either a specific relationship (such as friendship and kinship) between a pair of actors or define a particular action which an actor performs. Different kinds of ties specify different networks and are typically assumed to function differently. For example, the ties in a family network are distinctive from those in a working network, and the centrality in the “who loves whom” network obviously has different meaning than the centrality in the “who hates whom” network.

Social network theory is based on the intuitive notion that the social interaction patterns are essential to the individuals who reflect them. Network theorists believe that how individuals behave largely depends on how they interact with others and how they are tied to a social network. Furthermore, besides individual behavior, network theorists believe that the success or failure of societies and organizations often depends on the internal interaction pattern (Freeman, 2004).

Besides the theoretical essence, social network theory is also characterized as a distinctive methodology encompassing

techniques for data collection, statistical analysis, and visual representation. This approach is usually called social network analysis and will be discussed in the research methodology section. This paper draws on the social network theory to study the interaction pattern of OSS development project.

## RESEARCH METHODOLOGY

### Social Network Analysis

Social network analysis is used in our study to investigate the interaction pattern of the OSS development process. Social network analysis focuses on uncovering the interaction pattern of interdependent individuals (Freeman, 2004). Through a structural analysis of a social network diagram, a map depicting actors as well as the ties that connect them, social network analysis can reveal the patterns of relationships and the relative position of individuals in a specific social setting. This approach has been effectively used in organizational research, social support, mental health, and the diffusion of information (Freeman, 2004).

Social network analysis is used in our study for two primary reasons. First, the purpose of social network analysis fits our research objective. Social network analysis aims to analyze the relationship among a set of actors instead of their internal attributes. Our research aims to reveal the interaction pattern of OSS project teams. Therefore, social network analysis is helpful in answering our research questions.

Second, the rich interactive data extracted from OSS projects presents a “gold mine” for social network analysis. Social network analysis is grounded in the systematic analysis of empirical data. However,

there is usually a lack of convenient and objective resources from which to draw the links (i.e., relationships) among actors. Most OSS projects have online mailing lists, forums, and tracking systems that are open to public, thus providing a rich set of longitudinal data. Based on these public data, researchers are able to capture input data sets for social network analysis.

### Longitudinal Data

Because we are interested in studying how the interaction pattern of OSS projects evolves over time, cross-sectional observations of interaction networks are not sufficient. Cross-sectional observations of social networks are snapshots of interactions at a point in time and cannot provide traceable history, thus limiting the usefulness of the results. On the other hand, longitudinal observations offer more promise for understanding the social network structure and its evolvement. In this study, we extracted longitudinal data on OSS projects.

### Case Selection

OSS projects were selected from the SourceForge<sup>1</sup>, which is the world’s largest Web site hosting OSS projects. SourceForge provides free tools and services to facilitate OSS development. At the time of the study, it hosted a total of 99,730 OSS projects and involved 1,066,589 registered users (This data was retrieved on May 4, 2005). Although a few big OSS projects have their own Web sites (such as Linux), SourceForge serves as the most popular data resource for OSS researchers.

Following the idea of theoretical sampling (Glaser & Strauss, 1967), three OSS projects were selected from SourceForge in terms of their similarities and differences. Theoretical sampling requires theoretical relevance and purposes (Orlikowski, 1993).

In terms of relevance, the selection ensures that the interaction pattern of OSS projects over time is kept similar. Therefore, the projects that are selected have to satisfy two requirements. First, the projects must have considerable interaction among members during the development process. All three projects had more than 10 developers, and the number of bugs reported was more than 1,000. Second, since we are interested in the interaction over time, the projects must have a relatively long life. In our case, all three projects were at least three years old.

In addition to similarities, differences are sought among cases because the study aims to study interaction patterns of various OSS projects. Therefore, the three projects differ on several project characteristics, such as project size, project type, and intended audience. These differences

enable us to make useful contrasts during data analysis.

The Table 1 summarizes the three projects with a brief description.

### Data Collection and Analysis

Social network analysis can be divided into the following three stages (Borgatti, 2002).

1. Data collection. In this stage, researchers collect data, using surveys and questionnaires, or from documents and other data resources, and generate input data sets for social network analysis.
2. Statistical analysis. Based on mathematics algorithms, this stage generates network indices concerning group structure (such as centralization and

*Table 1. Summary of three projects*

		Net-SNMP	Compiere ERP + CRM	J-boss
Description		Net-SNMP allows system and network managers to monitor and manage hosts and network devices.	Compiere is a smart ERP+CRM solution covering all major business areas—especially for small-medium enterprises.	JBoss is a leading open source Java application server. After Linux and Apache, it is the third major open source project to receive widespread adoption by corporate IT.
Similarities	Bug reports (more than 1,000 bugs)	1,361	1,695	2,296
	Development duration (more than 3 years)	55 months (registered on 10/2000)	47 months (registered on 6/2001)	50 months (registered on 3/2001)
Differences	Software type	Internet, network management	Enterprise: ERP+CRM	J2EE-based middleware
	Group size (number of developers)	Small (14)	Median (44)	Large (75)
	Intended audience	Developers, system administrators	Business	Developers, system administrators

*Note: Data was retrieved in April 2004 from SourceForge.net.*

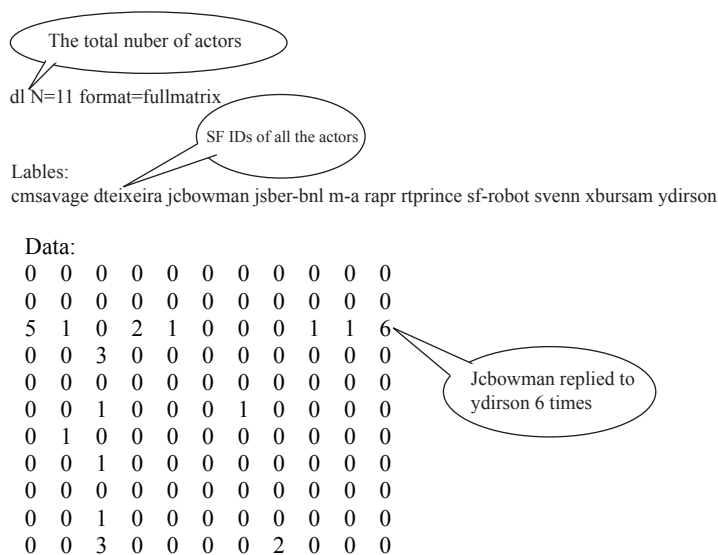
- density) as well as individual cohesion (such as centrality and bridges).
3. Visual representation. This stage employs network diagrams to show the interaction structure as well as the position of specific actors.

First is the data collection. The data were collected in April 2005 from SourceForge.net. Data were extracted from the bug tracking system of each project. We chose the bug tracking system as the primary data resource for three reasons. First, open source software is characterized as peer review of open codes. Raymond (1998) proposed the "Linus' law" in his well-known essay *The Cathedral and the Bazaar*—"Given enough eyeballs, all bugs are shallow." Therefore, the bug system can be viewed as the representative of open source spirit. Second, compared to other development activities, such as patch posting and feature request,

the bug-fixing process is the most active procedure to illustrate the close collaboration between developers and users as well as among developers themselves. Finally, the bug tracking system provides rich data that record the interactive process.

A Web spider program, which is based on the work of Crowston and Howison (2005) with necessary revision, was used to download the bug tracking Web pages from the project Web site. After that, a Web parsing program was developed to analyze the Web pages. The interaction data was extracted from the bug tracking Web pages month-by-month, starting from the date the project was registered until the date the data was downloaded for this study. The output of this stage is a social matrix describing the interaction among users. Figure 2 shows an example of such a social matrix for an OSS project. In the matrix, each row or column represents a distinctive participant,

Figure 2. An example of the social matrix for an OSS project





which is identified by a unique SourceForge user identity. The values of cells indicate the degree of the interaction between each pair of participants, which is counted by the amounts of messages that participant A (i.e., row A) replied to participant B (i.e., column B).

Second is the statistical analysis. Our study focuses on two important and distinctive properties of network structure—group centralization and core/periphery fitness. Ucinet, which was developed by Borgatti, Everett, and Freeman (2002), was used to calculate these two properties.

Group centralization, as suggested by Wasserman and Faust (1994), refers to the extent to which a network revolves around a single center. A typical case of centralized structure is a “star” network. Group

centralization can be viewed as a rough measure of inequity between individual actors, and the variability and dispersion of the interaction pattern.

The other property is core/periphery fitness. It measures the extent to which the network is close to a perfect core/periphery structure. The core/periphery structure depicts a dense, connected group surrounded by a sparse, unconnected periphery. The opposite structure is clique, which represents a structure of multiple subgroups, each with its own core and peripheries (Borgatti, 2002).

Finally is the visual representation. We used Ucinet (Borgatti et al., 2002) to draw the interaction networks for each of the three projects.

*Table 2. Three snapshots for each project*

		Net-SNMP	Compiere	JBoss
Group centralization (%)	1 <sup>st</sup> .	9.420	15.624	4.931
	2 <sup>nd</sup> .	3.071	2.294	4.45
	3 <sup>rd</sup> .	2.316	1.288	4.12
Core/periphery fitness	1 <sup>st</sup> .	0.674	0.774	0.485
	2 <sup>nd</sup> .	0.654	0.796	0.477
	3 <sup>rd</sup> .	0.651	0.765	0.501
Density	1 <sup>st</sup> .	0.0235	0.0584	0.0073
	2 <sup>nd</sup> .	0.0109	0.0610	0.0039
	3 <sup>rd</sup> .	0.0072	0.0571	0.0026
Average distance	1 <sup>st</sup> .	2.546	2.711	3.438
	2 <sup>nd</sup> .	2.794	2.302	3.281
	3 <sup>rd</sup> .	2.917	2.278	3.239
Distance-based cohesion	1 <sup>st</sup> .	0.181	0.198	0.118
	2 <sup>nd</sup> .	0.143	0.253	0.147
	3 <sup>rd</sup> .	0.141	0.279	0.136

## RESEARCH RESULTS

### Snapshots of the Three Projects

Monthly data were extracted from the bug tracking system of each project. To illustrate the trend of interaction pattern, we provide three snapshots for each project (see Figures 3-5)<sup>2</sup>.

Table 2 summarizes the relevant network characteristics of each project. In addition to the group centralization and core/periphery fitness, we also report other network characteristics, such as density, average distance, and distance-based cohesion. Density depicts how “close” the network looks, and it is a recommended measure of group cohesion (Blau, 1977; Wasserman & Faust 1994). The value of

density ranges from 0 to 1. Average distance refers to average distance between all pairs of nodes (Borgatti, 2002). Distance-based cohesion takes on values between 0 and 1—the larger the values, the greater the cohesiveness.

Looking at the statistical results and the network plots, we can observe the following.

First, the evolvement of interaction patterns of the three projects reveals a general trend. As shown in the network plots (i.e., Figures 3-5), the interaction pattern develops from a centralized one with a single (sometimes dual) hub with several distributed nodes, to a core/periphery structure that has a core (a group of core developers) together with several

Figure 3. Interaction patterns of Net-SNMP

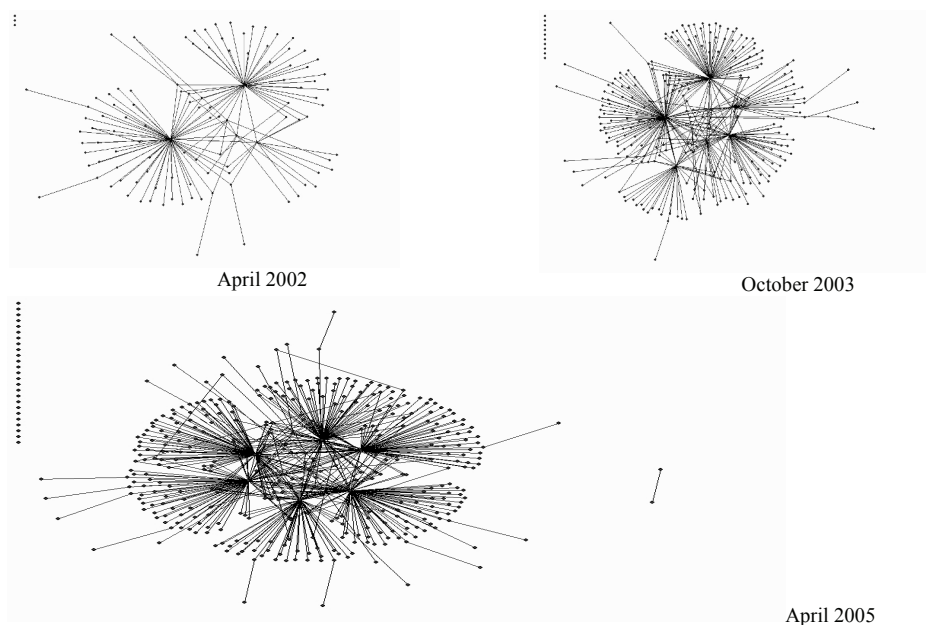


Figure 4. Interaction patterns of compiere CRM+ERP

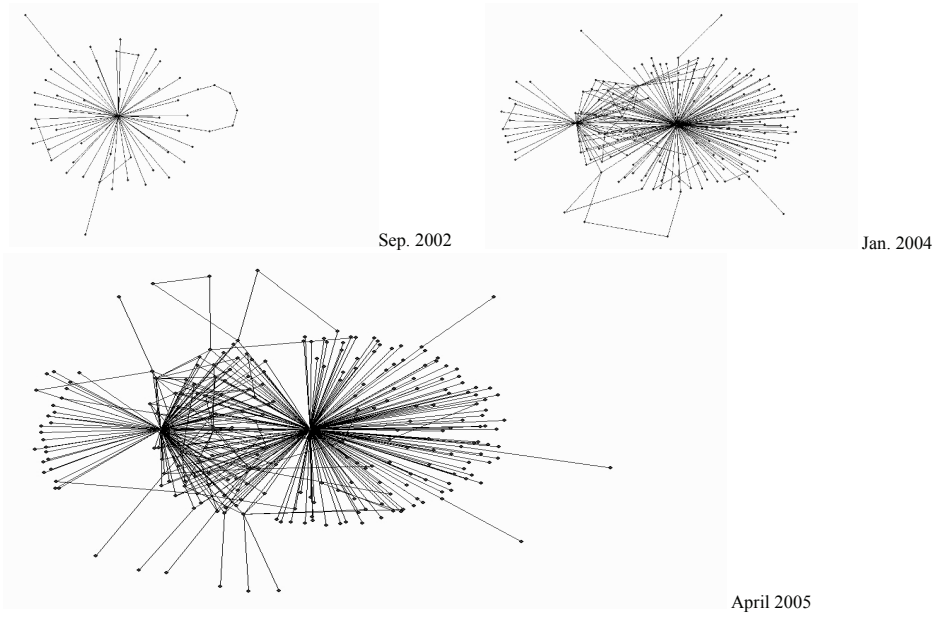
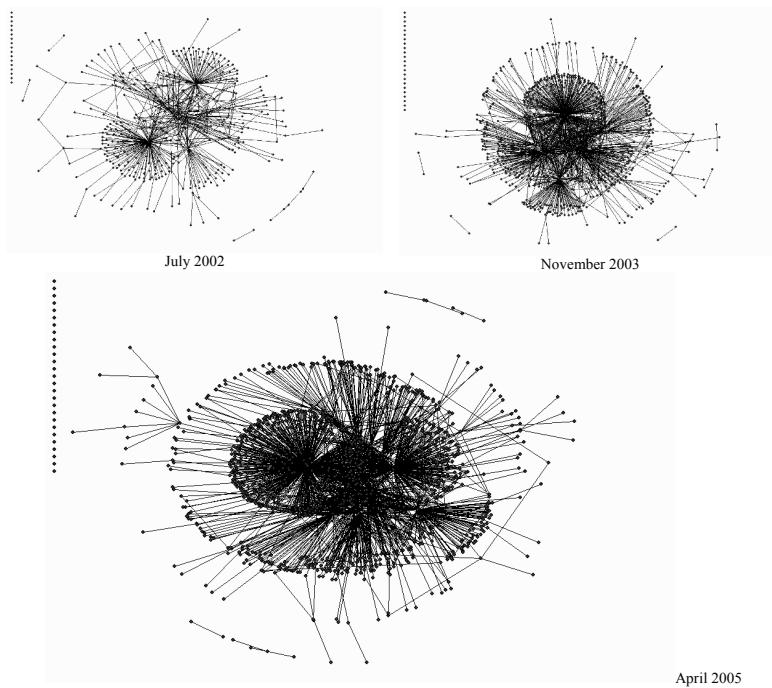


Figure 5. Interaction patterns of JBoss



hangers-on (periphery). Intense interactions exist within the core (among several core developers) and between each core member and his/her periphery. However, only loose relationships exist among peripheries. This pattern suggests a layer structure (i.e., core with its periphery) instead of a complete flat one with equal positions across all the members.

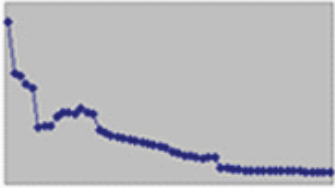
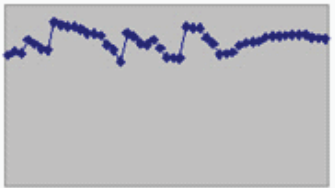
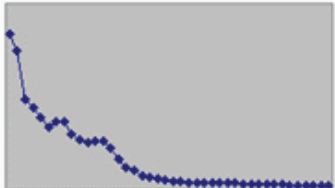
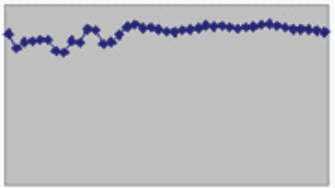

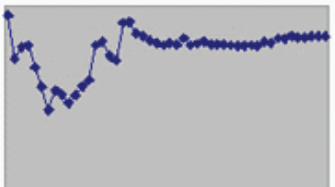
Second, although the interaction patterns of the three projects share some commonalities, their exact shapes are different. The shape of Net-SNMP (as shown in Figure 3) is more like a typical core/periphery compared to the other two. Compiere (as shown in Figure 4) keeps two

cores, and the shape looks like a dumbbell. JBoss (as shown in Figure 5), which is the largest project among the three, maintains a more complex structure that shows multiple layers instead of just one core with the rest as peripheries (e.g., Net-SNMP)

Third, as time goes by, the group centralization decreases across the three projects, showing a trend that moves from a centralized structure to a decentralized structure irrespective of project sizes (The three projects with different project sizes are shown in Table 1), project types, and intended audience.

Fourth, the indices of core/periphery fitness of each project fluctuate slightly but

Table 3. Group centralization and core/periphery fitness based on longitudinal data

	Group centralization	Core/periphery fitness
Net-SNMP		
Compiere		
JBoss		

maintain a relatively high value (larger than 0.5 on average). However, no observable trend was found across projects.

Fifth, since each project has a relatively large group (i.e., more than 100 actors including all the registered users), the values of density are relatively low with little variation. Therefore, density is not appropriate for comparing the projects.

From the snapshots, we observed the following trend. First, the OSS interaction network evolves into a core/periphery structure. Second, group centralization decreases over time. Third, core/periphery fitness stays relatively stable. To verify our observations, we used longitudinal data generated from the bug tracking systems to analyze the evolvement of interaction pattern (discussed in the following section).

### **Group Centralization and Core/Periphery Fitness**

Table 3 shows the values of both group centralization and core/periphery fitness over time based on the monthly interaction data. For each figure, the Y-axis indicates the social structure indices (i.e., group centralization or core/periphery fitness), and the X-axis reflects the time dimension.

Two primary observations can be made based on the statistical analysis.

First, the group centralization shows a decreasing trend across the three projects. This observation indicates that as OSS projects progress, the social network structure evolves from centralized to decentralized and then stabilizes. Also, the three figures suggest no substantial differences in the trend among the three projects.

Second, the core/periphery index is maintained at a relatively stable level for each project over time. In addition, the average fitness value stays relatively high for each project (larger than 0.5), indicat-

ing a closeness to a perfect core/periphery structure.

Besides a holistic view of network structure for OSS projects, the results also reveal other interesting findings. For example, by examining the core members over time, we found a relatively stable core for each project. The cores are usually project developers and administrators. This observation further demonstrates the existence of strong and stable core as well as loose hangers-on in OSS projects.

## **DISCUSSION**

This research uses the longitudinal data of three OSS projects selected from SourceForge to study the social network structures of OSS teams. The purpose of this study is to investigate the evolvement of interaction patterns of OSS project teams. The research results suggest a decrease of group centralization over time and a tendency of core/periphery structure in OSS project teams.

The network plots (as shown in Figures 3-5) indicate a layer structure instead of a flat one as suggested by earlier literature. The interaction pattern evolves from a single hub to a core/periphery structure. As the number of participants increases, a core with only one person (who may be the starter/initiator of the project) cannot satisfy the increasing requirements of development and communication. Therefore, other developers or active users join the core to serve as key members of the project. This results in a more stable structure, and the project is less dependent on a single leader.

With the growth of a software project, more people are attracted to the project. The original leader may not be able to solve all the technical problems encountered in the development process. Each key member has

his/her specialty, is responsible for solving relevant problems, and has his/her own periphery in the network plot. Although there are multiple peripheries in the project, collaboration among key members in the project is vital. This phenomenon of distribution and collaboration can be viewed as a critical success factor of OSS development. And the evolvement is vividly demonstrated in our social network analysis.

In a way, the social structure of OSS projects is both centralized and decentralized. On one hand, it is centralized in the sense that there is a core that consists of key members. These key members are responsible for various issues encountered during the development process. On the other hand, it is decentralized in the sense that the decision or communication core is not concentrated on one or two members but a group of key members.

Like any other research, this research has its share of limitations. First, the cases were only selected from SourceForge.net. Although SourceForge is the world's largest Web site hosting open source software, there are also some other similar Web sites. Therefore, the total number of OSS projects in SourceForge cannot be viewed as the whole population. However, as we argued before, SourceForge is probably the best data collection site for this research.

Second, the bug tracking system was chosen as our data resource. The selection of bug tracking system as our research setting and data resource may have had an effect on the outcome and results. Besides the bug tracking forum, there are other forums that also provide space for participants to communicate with one another, such as mailing lists and feature requests. However, as we highlighted earlier, the bug systems are the most active forum, providing rich interaction data. The bug tracking systems also

represent the spirit of open source software development. Examining the interaction data from other forums can be one of our research extensions in the future.

Third, because our research objective is to investigate interaction pattern, we chose projects that have a relatively large number of developers, a large number of bug reports, and relatively long history. Although we tried to involve different types of projects (i.e., different project sizes, project types, and intended audience), these three cases may not be representatives of OSS projects, for example, small projects with only one or two developers and few interactions. Increasing the sample size and including various types of OSS projects is one of our future research directions.

## IMPLICATIONS AND CONCLUSION

This paper examines the interaction patterns of OSS teams. The research findings suggest that the interaction structure starts from a single hub and evolves to a core/periphery model. We argue that the social structure of OSS teams is both centralized and decentralized. It is centralized in the sense that there exists a relatively stable core that consists of a group of key developers. It is also decentralized because of distributed decision making among key developers and the broad collaboration between developers and users as well as among developers themselves.

The paper presents the evolvement of the social structure of OSS projects from a longitudinal perspective. It also provides empirical evidence of the change of interaction patterns from a single hub to a core/periphery model. Moreover, the paper utilizes social network analysis as the research method. This approach has

been shown in this research as an effective tool in analyzing the social structure in OSS teams.

Social structure is an important variable for understanding social phenomenon. Open source software, with its open and unique nature, attracts researchers to ask a series of questions. For example, how do participants of OSS projects interact and collaborate with each other? What factors facilitate the interaction and the collaboration? And further, how does the collaboration affect project performance of OSS teams? Social network analysis is a good approach to investigate these questions. This research represents a pioneering effort in this direction.

## REFERENCES

- Ahuja, M., & Carley, K. (1999). Network structure in virtual organizations. *Organization Science*, 10(6), 741-747.
- Blau, P. M. (1977). *Inequity and heterogeneity*. New York: Free Press.
- Borgatti, S. (2002). *Basic social network concepts*. Retrieved from <http://www.analytictech.com/networks/basic%20concepts%202002.pdf>
- Borgatti, S. P., Everett, M. G., & Freeman, L. C. (2002). *Ucinet for Windows: Software for social network analysis*. Harvard, MA: Analytic Technologies.
- Crowston, K., & Howison, J. (2005). The social structure of free and open source software development. *First Monday*, 10(2).
- Erickson, J., Lyytinen, K., & Siau, K. (2005). Agile modeling, agile software development, and extreme programming: The state of research. *Journal of Database Management*, 16(4), 88-100.
- Freeman, L. C. (2004). The development of social network analysis: *A study in the sociology of science*. Vancouver, Canada: Empirical Press.
- Gacek, C., & Arief, B. (2004). The many meanings of open source. *IEEE Software*, 21(1), 34-40.
- Glaser, B. G., & Strauss, Anselm L., (1967). *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Chicago, Aldine Publishing Company
- Hann, H., Robert, J., & Slaughter, S. (2004). Why developers participate in open source software projects: An empirical investigation. In *Twenty-Fifth International Conference on Information Systems*, (pp. 821-830). Washington, DC: .
- Hemetsberger, A. (2004). Sharing and creating knowledge in open-source communities: The case of KDE. *The Fifth European Conference on Organizational Knowledge, Learning, and Capabilities in Innsbruck, Austria*.
- Jørgensen, N. (2001). Putting it all in a trunk: Incremental software development in the freeBSD open source project. *Information Systems Journal*, 11, 321-336.
- Krebs, V., & Holley, J. (2004). *Building sustainable communities through network building*. Retrieved from <http://www.orgnet.com/Building-Networks.pdf>
- Lee, G. K., & Cole, R. E. (2003). From a firm-based to a community-based model of knowledge creation: The case of the Linux kernel development. *Organization Science*, 14(6), 633-649.
- Madey, G., Freeh, V., & Tynan R. (2002). The open source software development phenomenon: An analysis

- based on social network theory (AM-CIS2002). Dallas, TX.
- Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2000). A case study of open source software development: The Apache server. ICSE 2000.
- Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, 11(3), 309–346.
- Moon, J. Y., & Sproull, L. (2000). Essence of distributed work: The case of Linux kernel. *First Monday*, 5(11).
- Orlikowski, W. J. (1993). CASE tools as organizational change: investigating incremental and radical changes in systems development. *MIS Quarterly*, 17(3), 309-340.
- Raymond, E. S. (1998). The cathedral and the bazaar. *First Monday*, 3(3), Retrieved January, 2005, from <http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/>
- Sagers, G. W. (2004). The influence of network governance factors on success in open source software development projects. In *Twenty-Fifth International Conference on Information Systems* (pp. 427-438). Washington, DC:
- Schaefer, R. T., & Lamm, R. P. (1998). *Sociology* (6<sup>th</sup> ed.). McGraw-Hill.
- Scott, J. (2000). *Social network analysis. A handbook* (2<sup>nd</sup> ed.). London: SAGE Publications.
- Siau, K., & Cao, Q. (2001). Unified modeling language—A complexity analysis. *Journal of Database Management*, 12(1), 26-34.
- Siau, K., Erickson, J., & Lee, L. (2005). Theoretical versus practical complexity: The case of UML. *Journal of Database Management*, 16(3), 40-57.
- Siau, K., & Loo, P. (2006). Identifying difficulties in learning UML. *Information Systems Management*, 23(3), 43-51.
- Stamelos, I., Angelis, L., Oikonomu, A., & Bleris, G. L. (2002). Code quality analysis in open-source software development. *Information Systems Journal*, 12(1), 43-60.
- Von Hippel, E., & Von Krogh, G. (2003). Open source software and the „private-collective“ innovation model: Issues for organization science. *Organization Science*, 14, 209-223.
- Von Krogh, G., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: A case study. *Research Policy*, 32(7), 1217-1241.
- Wasserman, S., & Faust, K., (1994). *Social Network Analysis: Methods and Applications*. New York: Cambridge University Press.

#### ENDNOTE

- <sup>1</sup> The Web address for SourceForge is [www.sourceforge.net](http://www.sourceforge.net).
- <sup>2</sup> The three time stamps for Net-SNMP are 4/2002, 10/2003 and 4/2005; for Compiere are 9/2002, 1/2004 and 4/2005; and for Jboss are 7/2002, 11/2003 and 4/2005.



*Yuan (Yoanna) Long is an assistant professor in Hasan School of Business at the Colorado State University-Pueblo. Her research areas include virtual group, social network, open source software development, and knowledge management. She has published in Journal of Global Information Management, Industrial Management & Data Systems, and Annals of Cases on Information Technology.*

*Keng Siau is the E. J. Faulkner Professor in management information systems (MIS) at the University of Nebraska, Lincoln (UNL). He is the director of the UNL-IBM Institute, editor-in-chief of the Journal of Database Management, and co-editor-in-chief of the Advances in Database Research series. He received his PhD degree from the University of British Columbia (UBC), where he majored in management information systems and minored in cognitive psychology. His master's and bachelor degrees are in Computer and Information Sciences from the National University of Singapore. Dr. Siau has over 200 academic publications. He has published more than 85 refereed journal articles, and these articles have appeared (or are forthcoming) in journals such as Management Information Systems Quarterly, Communications of the ACM, IEEE Computer, Information Systems, ACM SIGMIS's Data Base, IEEE Transactions on Systems, Man, and Cybernetics, IEEE Transactions on Professional Communication, IEEE Transactions on Information Technology in Biomedicine, IEICE Transactions on Information and Systems, Data and Knowledge Engineering, Decision Support Systems, Journal of Information Technology, International Journal of Human-Computer Studies, International Journal of Human-Computer Interaction, Behaviour and Information Technology, Quarterly Journal of Electronic Commerce, and others. In addition, he has published more than 90 refereed conference papers (including 10 ICIS papers), edited/co-edited more than 10 scholarly and research-oriented books, edited/co-edited 9 proceedings, and has written more than 15 scholarly book chapters. He served as the Organizing and Program Chairs of the International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD) (1996–2005).*

Reproduced with permission of copyright owner. Further reproduction prohibited without permission.